UNIVERZA V LJUBLJANI

FAKULTETA ZA RAČUNALNIŠTVO IN INFORMATIKO

Neža Štrukelj

Uvajanje metodologije Scrum na večjem projektu: študija primera in analiza faktorjev sprejemljivosti

MAGISTRSKO DELO

MAGISTRSKI PROGRAM
INFORMACIJSKI SISTEMI IN ODLOČANJE

MENTOR: prof. dr. Viljan Mahnič

Ljubljana, junij 2016

Univerza v Ljubljani
Fakulteta za računalništvo in informatiko

Večna pot 113
1000 Ljubljana, Slovenija
telefon: 01 47 98 100
www.fri.uni-lj.si
e-mail: dekanat@fri.uni-lj.si

Številka: 167-MAG-ISO/2016
Datum: 06. 04. 2016

Neža ŠTRUKELJ, prof. mat in rač.

Ljubljana

Fakulteta za računalništvo in informatiko Univerze v Ljubljani izdaja naslednjo magistrsko nalogo

Naslov naloge: **Uvajanje metodologije Scrum na večjem projektu: študija primera in analiza faktorjev sprejemljivosti**

Introducing Scrum into a large project: a case study and analysis of the acceptance factors

Tematika naloge:
Zahteve naročnikov pri razvoju informacijskih sistemov se danes pogosto in hitro spreminjajo, saj morajo slediti trgu oziroma potrebam svojih uporabnikov. Temu mora slediti tudi razvoj informacijskih sistemov. Sodobni trendi priporočajo agilne metodologije, ki omogočajo iterativen in inkrementalen razvoj ter se prilagajajo spremembam uporabnikovih zahtev.
V magistrskem delu prikažite izkušnje z uporabo metodologije Scrum na primeru obsežnejšega realnega projekta. Za ta projekt izdelajte ti. postmortem analizo ter opišite njegove pozitivne in negativne plati. S pomočjo ustrezne ankete in intervjujev s člani projektne skupine analizirajte stopnjo sprejetja posameznih praks in dolgoročno sprejemljivost metodologije Scrum v podjetju, ki je izvajalo omenjeni projekt. Pri tem uporabite ustrezne modele iz literature (Kwon-Zmudov model stopenj sprejemanja tehnoloških inovacij in Rogersovo teorijo difuzije inovacij). Na podlagi rezultatov podajte priporočila za izboljšanje uporabe metodologije Scrum v praksi.

Mentor:
izr. prof. dr. Viljan Mahnič

Dekan:
prof. dr. Nikolaj Zimic

Hvala Patriciju za spodbudo, razumevanje in veliko podporo pri pisanju.

Hvala sinovoma, 5-letnemu Oskarju in 3-letnemu Marku, ki sta bila v času pisanja tega dela zelo pogumna in skušala razumeti, da mami piše 'knjigo za šolo'.

Hvala Sanji, ki je delila svoje znanje in navdušenje nad Scrumom ter me informirala o začetku projekta.

Hvala sijajni ekipi 'E2', s katero sem imela priložnost sodelovati in prakticirati Scrum na realnem primeru.

Hvala mentorju, prof. dr. Viljanu Mahniču, za odzivnost, konstruktivne kritike in usmerjanje pri izdelavi dela.

Oskarju in Marku

Logika te pripelje od točke A do točke B. Domišljija te pripelje povsod.
Albert Einstein

KAZALO

KAZALO TABEL

KAZALO SLIK

SEZNAM UPORABLJENIH KRATIC

CMS	*ang. Content Management System* sistem za upravljanje vsebin
DOI	*ang. Diffusion of Inovation theory* teorija difuzije inovacije
IT	*ang. Information Technology* informacijska tehnologija
MRVL	Matični register vozil in prometnih listin
PRS	Poslovni register Slovenije
PV	produktni vodja
PZI	projekt za izvedbo
R	razvijalec
RC	raziskovalni cilj
SM	skrbnik metodologije
SOA	*ang. Service-Oriented Architecture* storitveno usmerjena arhitektura

POVZETEK

Zahteve naročnikov pri razvoju informacijskih sistemov se danes pogosto in hitro spreminjajo, saj morajo slediti trgu oziroma potrebam svojih uporabnikov, temu pa mora slediti tudi razvoj informacijskih sistemov. Sodobni trendi priporočajo agilne metodologije, ki omogočajo iterativen in inkrementalen razvoj ter se prilagajajo spremembam zahtev v času razvoja.

Naloga predstavlja študijo primera, v kateri analiziramo uvedbo metodologije Scrum na projektu, ki ga je za potrebe javne uprave izvajalo eno izmed večjih slovenskih podjetij za razvoj programske opreme.

V tem primeru je šlo za prvo uporabo Scruma, zato je na eni strani posebna pozornost posvečena stopnji sprejetja posameznih praks, to je, do katere mere razvijalci obvladajo posamezne prakse po koncu projekta, in na drugi strani faktorjem, ki po mnenju razvijalcev najbolj vplivajo na uspešno uvedbo Scruma in njegovo dolgoročno sprejemljivost ter analizi pozitivnih in negativnih izkušenj/vidikov.

Pri analizi stopnje sprejetja posamezne prakse smo se oprli na 6-stopenjski Kwon-Zmud-Cooperjev model sprejetja inovacije; pri identifikaciji najpomembnejših faktorjev na Rogersovo teorijo difuzije inovacije, za raziskavo pozitivnih in negativnih vidikov pa smo izvedli postmortem analizo. Dobljene rezultate smo primerjali še s sorodno študijo o uporabi Scruma v nemški zavarovalnici.

Potrebne podatke smo zbrali z vprašalnikom, ki je bil sestavljen iz petih delov. V prvem delu so bila splošna vprašanja o anketirancu; v drugem so se vprašanja nanašala na stopnje sprejetja posameznih praks; v tretjem delu smo spraševali o faktorjih, ki vplivajo na sprejemljivost Scruma; četrti del je vseboval vprašanja za primerjavo s hipotezami Overhage-Schlaudererja in v zadnjem delu so bila vprašanja, ki so se nanašala na značilnosti projekta. Za potrditev oziroma razjasnitev rezultatov vprašalnika smo izvedli še delno strukturirane intervjuje.

Ugotovili smo, da je nekatere stvari potrebno izboljšati, nekatere pa obdržati. Podali smo priporočila za izboljšanje izbire članov skupine Scrum, vključitev naročnika v proces Scruma, sprotno spremljanje poteka dela in ocenjevanje uporabniških zgodb. Stvari, ki jih je smiselno

negovati tudi v prihodnje, so skupni prostori in sodobna infrastruktura, osebna in strokovna rast posameznika ter možnost dela po metodologiji Scrum tudi v prihodnje.

Ključne besede: Scrum, agilni razvoj programske opreme, model sprejetja inovacije, teorija difuzije inovacije, DOI, postmortem analiza.

ABSTRACT

Customers' requirements are changing often and quickly in the information system development nowadays accommodate to user' needs and market change. Information system development should reflect this constant change too. Current trends in information system development recommend agile methodologies that enable iterative and incremental development and are adaptable to changes in requirements during development.

This work presents a case study analysis of the introduction of the Scrum methodology on a public administration project carried out by one of the major Slovenian software development companies.

This case was the first use of the Scrum methodology by the software development company. Therefore, special attention is paid, firstly, to the level of adoption of Scrum practices, which means the stage to which the developers assimilated each practice after the end of the project; and secondly to the factors with greatest impact on the successful introduction of Scrum and its long-term acceptability according to developers and analysis of the positive and negative experiences / aspects.

To analyze the adoption stage of each practice, we used the 6-stages Kwon-Zmud-Cooper Innovation assimilation stages model. To identify the most important factors, we used the Rogers' Diffusion of innovation theory. To study the positive and negative aspects, we performed a postmortem analysis. The results were compared with a similar study on the use of Scrum in a German insurance company.

The data was collected by a questionnaire, which consisted of five parts. The first part included the general questions about the respondent. The questions in the second part referred to the degree of adoption of certain practices. In the third part we asked about the factors which influence long-term Scrum acceptance. The fourth part contained questions needed for a comparison with Overhage-Schlauderer hypotheses and the last part of the questionnaire included questions relating to the characteristics of the project. To confirm or clarify the results of the questionnaire semi-structured were carried out.

It was established that some things need to be improved and others to be kept and maintained. Therefore, recommendations are provided for improving the selection process of the Scrum

group members, the client inclusion in the Scrum process, monitoring workflow and user stories estimation. Things that should be nurtured in the future are a joint office and modern infrastructure, individual personal and professional growth and the possibility to work with Scrum methodology in the future.

Keywords: Scrum, agile software development, Innovation assimilation stages, Diffusion of innovation theory, DOI, postmortem analysis.

1 UVOD

Razvoj programske opreme je kompleksen proces in veliko projektov se konča z zamudo, ne izpolni prvotnih ciljev projekta ali pa so celo preklicani. Postmortem analize teh projektov so znova in znova pokazale enake napake: od slabo podanih zahtev, nezainteresiranih končnih uporabnikov, napačnih začetkov, dolgega cikla dostave programske opreme, do pretirano podrobnih postopkov [33]. Poleg tega se danes tehnologije in poslovne zahteve spreminjajo v času projekta. Proces razvoja programske opreme je bil v središču pozornosti zanimanja mnogih vodij, inženirjev in raziskovalcev prav zaradi velikega deleža napak v industriji programske opreme [6].

Alternativa tradicionalni metodologiji razvoja programske opreme, ki ji očitajo preveč birokracije in neobvladovanja stalnih sprememb zahtev med razvojem, so agilne metodologije. Agilne metodologije poleg prihrankov denarja, krajših dostavnih časov izdelkov, prilagajanja spremembam in kakovostnejših aplikacij prinašajo tudi večje zadovoljstvo razvojne skupine [19, 27]. Agilne metodologije se vedno bolj uveljavljajo, med njimi prevladuje Scrum [21, 22, 23], ki se po vprašalniku VersionOne uporablja kar v 70 % [32], zato so izkušnje Scruma vredne kritične analize. Scrum predstavlja nov način dela, ki bistveno spremeni način razmišljanja in dela razvijalcev programske opreme [3], zato se tudi v literaturi pojavljajo članki, ki opisujejo primere uvedbe Scruma v praksi [10].

V skladu s trendi razvoja informacijskih sistemov smo Scrum uporabili pri projektu prenove portala za državljane in zalednih sistemov. Projekt je bil obsežen in je trajal več kot dve leti. Z metodologijo Scrum se je večina članov skupine Scrum podrobneje spoznala prav na tem projektu, saj je bil za podjetje to prvi projekt, ki je bil izveden po tej metodologiji. Med samim projektom je bilo veliko sprememb, od menjav kadra na strani izvajalca kot tudi na strani naročnika do vsebinskih in tehničnih sprememb. Skupina Scrum je sprva imela velike težave pri sprejemanju praks metodologije Scrum, a je kljub vsem vzponom in padcem projekt uspešno pripeljala do konca in ob tem uspela zgraditi povezano in samoorganizirano skupino. Zadovoljstvo ob koncu projekta je bilo veliko tako na strani naročnika kot tudi izvajalca.

Motivacija dela je analizirati izkušnje končanega projekta. Na uspešnost projekta lahko gledamo tudi z vidika stopenj sprejetosti praks Scrum in stopenj difuzije faktorjev Scrum, zato smo si zastavili štiri raziskovalne cilje.

Prvi cilj magistrskega dela je ugotoviti, do katere stopnje so člani skupine Scrum usvojili posamezne tipične prakse Scrum in določiti stopnjo sprejetja praks Scrum po 6-stopenjskem modelu sprejetja inovacij Kwona-Zmuda-Cooperja [4, 9]. V ta namen smo definirali prvi raziskovalni cilj:

- **RC1**: Do katere stopnje je skupina usvojila tipične prakse Scrum?

Drugi cilj dela je identificirati tiste faktorje, ki vplivajo na dolgoročno sprejemljivost Scruma pri razvoju programske opreme. V literaturi obstajajo študije o sprejemljivosti ekstremnega programiranja [12], za Scrum pa ne, čeprav je Scrum po raziskavah sodeč daleč najbolj razširjena agilna metodologija [32]. Za ugotavljanje sprejemljivosti inovacije obstaja več modelov, na primer Rogersov *Diffusion of Inovation* [18] in Overhage-Schlaudererjev *Framework of Drivers and Inhibitors* [15]. Da bi raziskali, kateri so ti faktorji, smo osnovali drugi raziskovalni cilj, ki temelji na Rogersovem modelu [18]:

- **RC2**: Kateri faktorji pomembno vplivajo na odločitev za sprejem metodologije Scrum?

Tretji cilj je primerjava rezultatov naše raziskave z rezultati iz literature. Overhage in Schlauderer [14] sta s pomočjo intervjujev v večji nemški zavarovalnici preverjala osem hipotez, katerih osnova so trije Rogersovi faktorji sprejemanja inovacije. Za potrebe primerjave rezultatov smo definirali tretji raziskovalni cilj:

- **RC3**: Primerjava z rezultati znanstvene literature [14], ki se nanašajo na uvedbo Scruma.

Četrti cilj je izvesti postmortem analizo [1], katere fokus so negativni in pozitivni vidiki projekta s stališča Scruma. Nato je namen pridobljene rezultate analizirati in podati predlog izboljšave oziroma nadaljnje aktivnosti za izboljšanje uporabe Scruma v podjetju, s poudarkom na tistih praksah Scrum, ki so bile zaznane kot manj sprejemljive. V želji, da bi podjetje tudi v prihodnje razvijalo programske rešitve po metodologiji Scrum, je smiselno definirati še četrti raziskovalni cilj:

- **RC4**: Kakšna so priporočila podjetju za nadaljnjo uporabo Scruma?

Zgradba dela: v drugem poglavju predstavljamo metodologijo Scrum in modele, ki predstavljajo teoretično podlago za izdelavo študije.

V tretjem poglavju bralca podrobneje seznanimo z uvedbo metodologije Scrum na praktičnem primeru, to je prenova portala za državljane in zalednih sistemov.

V četrtem poglavju je prikazan načrt študije, s pomočjo katere želimo odgovoriti na zastavljene raziskovalne cilje. Podatke za analizo smo pridobili s pomočjo vprašalnikov, katerih struktura je v tem poglavju podrobneje predstavljena.

V petem poglavju so prikazani rezultati in analiza vprašalnikov, kar nam je predstavljalo osnovo za delno strukturirane intervjuje.

V šestem poglavju je predstavljena struktura delno strukturiranih intervjujev. Za vsak raziskovalni cilj smo določili vprašanja, ki so nam služila kot iztočnica za pogovor z intervjuvanci in nam hkrati omogočila primerjavo dobljenih rezultatov.

V sedmem poglavju analiziramo dobljene podatke intervjujev.

V osmem poglavju podamo priporočila podjetju tako za izboljšanje uporabe kot tudi za ohranjanje metodologije Scrum v prihodnosti.

V devetem poglavju zaključimo delo s sklepnimi ugotovitvami.

2 AGILNI RAZVOJ PROGRAMSKE OPREME

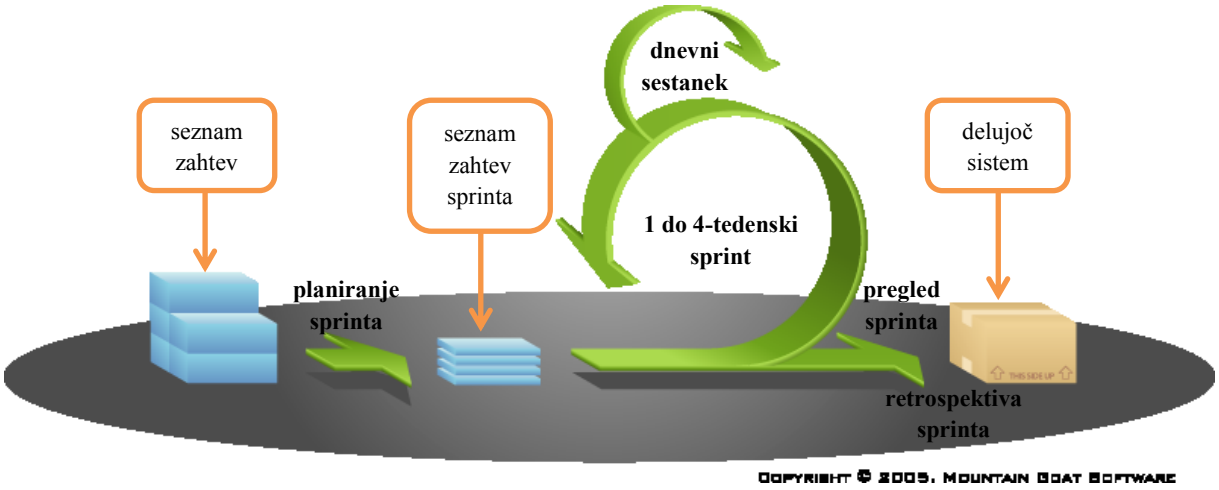
V obsežni raziskavi VersionOne [32] so ugotovili, da večina uporabnikov agilnih pristopov prihaja prav iz industrije razvoja programske opreme. Leta 2015 je na agilen način delalo 95 % vprašanih organizacij, v 43 % organizacijah večina skupin znotraj posamezne organizacije dela agilno. Število organizacij, ki se urijo v agilnosti, iz leta v leto narašča. Daleč najbolj uporabljena agilna metodologija vprašanih je Scrum [32]. Koncepti Scruma segajo v leto 1986, ko sta Takeuchi in Nonaka opisala, kako Honda, Canon in Fuji-Xerox dosegajo dobre rezultate s samoorganiziranimi skupinami [24].

2.1 METODOLOGIJA SCRUM

Scrum je agilni pristop za razvoj inovativnih izdelkov in storitev. Je ogrodje za organizacijo in vodenje dela, ne pa standardiziran postopek z načrtovanimi zaporednimi koraki, ki zagotavljajo uspeh in zadovoljstvo naročnika. Temelji na sklopu vrednot, načel in praks, ki zagotavljajo temelje. Vsaka organizacija doda svoje inženirske prakse in specifične pristope [19], pri tem pa mora skrbeti, da ohranja koristi, ki jih zagotavlja Scrum [17]:

- produkt je razdeljen na vrsto manjših obvladljivih delov,
- kljub spreminjajočim se zahtevam je viden napredek,
- transparentnost procesa,
- boljša komunikacija v skupini,
- skupina je odgovorna za uspeh,
- stranke sproti dobivajo delujočo kodo,
- stranke sproti dajejo povratno informacijo o delovanju produkta,
- boljša se odnos s stranko,
- ustvarjena je kultura, kjer vsakdo pričakuje, da bo projekt uspel.

Razvoj programske opreme rešuje kompleksne probleme, ki so nepredvidljivi, zato se procesa razvoja programske rešitve ne da natančno planirati [21]. Za reševanje takih problemov je potreben iterativni razvoj (ang. *iteration*), kjer vsaka iteracija prinaša delujočo celoto, ta pa predstavlja doprinos (ang. *increment*) trenutni rešitvi, kar prikazuje Slika 1. Za realizacijo takega procesa Scrum predvideva tri vloge.



Slika 1: Okvir Scrum

2.1.1 VLOGE

V skupini Scrum nastopajo tri različne vloge [17, 19]:

- **Produktni vodja** (ang. *product owner*) je odgovoren za vsebino seznama zahtev in določanje vrstnega reda razvijanja funkcionalnosti. Ob koncu sprinta uporabniške zgodbe sprejme ali zavrne.
- **Skrbnik metodologije** (ang. *ScrumMaster*) je odgovoren za vodenje skupine in kreiranje ter sledenje procesu dela, ki temelji na ogrodju, ki ga ponuja Scrum. Odgovoren je za zaščito skupine pred zunanjimi vplivi in ima vodilno vlogo pri odpravljanju ovir, ki zavirajo produktivnost skupine.
- **Razvojna skupina** (ang. *development team*) se nanaša na vsakogar, ki je na strani razvoja, pa naj gre za programerja, testerja, analitika, oblikovalca uporabniške izkušnje, skrbnika baze podatkov in tako dalje. Razvojna skupina običajno šteje od tri do devet razvijalcev [23] in skupaj morajo imeti vsa znanja, da izdelajo dober in delujoč izdelek. Naloga razvojne skupine je, da se samoorganizira tako, da bo v času sprinta uspešno končala zahteve, za katere se je dogovorila s produktnim vodjem.

Produktni vodja, skrbnik metodologije in razvojna skupina sestavljajo skupino Scrum (ang. *Scrum team*). Vsak član skupine mora razumeti svojo vlogo in zahteve vsake iteracije. Skupina mora imeti jasen fokus in prioritete morajo biti nedvoumne [17].

2.1.2 AKTIVNOSTI IN IZDELKI

Slika 1 prikazuje potek Scruma, ki je predstavljen v nadaljevanju. Na začetku projekta se ustvari seznam zahtev (ang. *product backlog*), ki se v času projekta dopolnjuje in spreminja.

Količina zahtev, ki so v seznamu zahtev, je prevelika za eno iteracijo, zato skupina pred začetkom iteracije načrtuje (ang. *sprint planning*), katere zahteve so prioritetne. Izbrane prioritetne zahteve predstavljajo seznam zahtev sprinta (ang. *sprint backlog*).

Delo je organizirano v kratkih, časovno omejenih iteracijah (ang. *sprint*), ki navadno trajajo od enega do štiri tedne. Med vsako iteracijo samoorganizirana skupina opravi vse delo, to je od načrta, izvedbe do testiranja, po potrebi tudi dokumentiranja, kar na koncu pripelje do končnega izdelka, ki je primeren za produkcijo [19].

V času sprinta se skupina srečuje na kratkih dnevnih sestankih (ang. *Daily Scrum*), ki so časovno omejeni do največ 15 min [19]. Na sestanku vsak član skupine pove, kaj je naredil od prejšnjega sestanka, kaj bo delal do naslednjega sestanka in ali je naletel na kakšne ovire, ki mu onemogočajo dokončanje zahteve. Ker je dnevni sestanek časovno omejen, se ovire samo osvetli, ne pa tudi išče rešitve zanje [17].

Skupina lahko zmanjša število planiranih zahtev v času iteracije, ne more pa spremeniti dneva konca sprinta, saj je dolžina sprinta fiksna [17]. Ob koncu iteracije, na pregledu sprinta (ang. *sprint review*), skupina skupaj z naročnikom pregleda izdelane funkcionalnosti. Cilj pregleda je pridobiti povratno informacijo naročnika in potrditi izdelane zgodbe.

Po pregledu sprinta sledi retrospektiva sprinta (ang. *sprint retrospective*), kar predstavlja zadnjo aktivnost sprinta. Retrospektiva je priložnost za pregled in prilagoditev procesa dela s poudarkom na stalnem izboljševanju procesa, da dobra skupina postane odlična.

Po vsaki iteraciji mora biti nova funkcionalnost taka, da je primerna za produkcijo. Če se funkcionalnosti ne izdajo po vsaki iteraciji, se naredi izdaja (ang. *release*), ki vsebuje funkcionalnosti več iteracij.

Ob zaključku vsake iteracije se celoten postopek ponovi z načrtovanjem nove iteracije, kjer se upošteva naročnikovo povratno informacijo s pregleda sprinta in njegove trenutne prioritete zahtev.

2.2 SCRUM KOT INOVACIJA

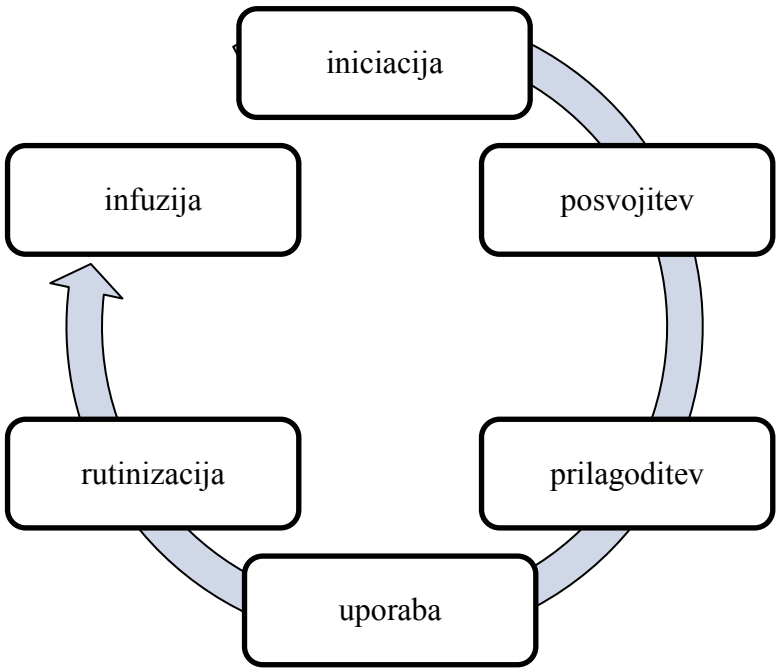
Scrum uvaja novo metodologijo za razvoj informacijskih sistemov. S teoretičnega vidika ga je mogoče opredeliti kot inovacijo [18], saj nove prakse bistveno spremenijo način razmišljanja in dela razvijalcev programske opreme, prav tako tudi celotnega podjetja [3, 14].

2.2.1 STOPNJE SPREJETJA INOVACIJE

Da bi opisali proces, v katerem podjetje, skupina ali posameznik sprejme inovacijo, so Kwon, Zmud in Cooper [4, 9] definirali 6-stopenjski model sprejetja inovacije (ang. *Innovation*

assimilation stages), ki ga prikazuje Slika 2. Vsaka stopnja opisuje raven, do katere je inovacija prodrla pri sprejemanju:

- **iniciacija** (ang. *Initiation*)
  Povezava med inovacijo in njeno uporabo s strani skupine, ki jo sprejema, je ugotovljena.
- **posvojitev** (ang. *Adoption*)
  Dosežena je odločitev za sprejem inovacije.
- **prilagoditev** (ang. *Adaptation*)
  Inovacija je prilagojena in pripravljena za uporabo v organizaciji; člani skupine so usposobljeni za njeno uporabo.
- **uporaba** (ang. *Acceptance*)
  Člani skupine se zavežejo k uporabi inovacije.
- **rutinizacija** (ang. *Routinization*)
  Uporaba inovacije postane običajna aktivnost.
- **infuzija** (ang. *Infusion*)
  Inovacija se uporablja na celovit način, s čimer se učinkovitost skupine povečuje.



Slika 2: 6-stopenjski model sprejetja inovacije

Po Senapathiju in Srinivasanju [20] se prve tri stopnje nanašajo na stanje pred usvojitvijo uporabe inovacije, medtem ko se zadnje tri stopnje nanašajo na stanje po usvojitvi uporabe inovacije. Da bi bili učinkoviti, bi morali pri uporabi inovacije doseči eno od stanj po usvojitvi.

2.2.2 TEORIJA DIFUZIJE INOVACIJE

Rogersova [18] teorija difuzije inovacije (ang. *The Diffusion of Inovation theory*), v nadaljevanju DOI, pojasnjuje, kako se inovacija širi znotraj družbenega sistema v daljšem časovnem obdobju. Končni rezultat difuzije je, da ljudje sprejmejo inovacijo, pa naj bo to ideja, obnašanje ali izdelek. Sprejetje pomeni, da oseba dela nekaj drugače, kot je delala prej.

Teorija DOI identificira pet skupin faktorjev (ang. *factors*), ki vplivajo na dolgoročno sprejemljivost inovacije [18]:

- **faktorji inovacije** (ang. *innovation factors*),
- **faktorji posameznika** (ang. *individual factors*),
- **faktorji naloge** (ang. *task factors*),
- **faktorji okolja** (ang. *environmental factors*),
- **faktorji organizacije** (ang. *organizational factors*).

Vsaka skupina vsebuje več faktorjev (ang. *traits*), kar kaže Tabela 1. Tako Rogersov model skupaj obsega 29 faktorjev [11, 13]. Mnoge od teh faktorjev merimo z zaznavanjem [13].

Tabela 1: Rogersove skupine faktorjev in posamezni faktorji

Skupina faktorjev	Faktor	Opis
Inovacija	**relativna prednost** (ang. *relative advantage*)	Inovacija predstavlja boljšo rešitev od tiste, ki je trenutno v uporabi.
	preprostost uporabe (ang. *ease of use*)	Stopnja, do katere je inovacija zaznana kot lahko razumljiva in uporabna.
	kompatibilnost (ang. *compatibility*)	Stopnja, do katere je inovacija zaznana kot skladna z obstoječim načinom dela, obstoječimi vrednotami, preteklimi izkušnjami in potrebami potencialnih posvojiteljev.
	možnost opazovanja, kako inovacijo uporabljajo drugi (ang. *visibility*)	V kolikšni meri so rezultati in način dela z inovacijo vidni drugim.
	možnost preizkušanja (ang. *trialability*)	Stopnja, do katere se lahko z inovacijo eksperimentira pred dokončno odločitvijo o ne/uporabi.
	strošek (ang. *price*)	Strošek inovacije.

Skupina faktorjev	Faktor	Opis
	razreševanje problema (ang. *problem solver*)	Želja po sprejetju inovacije je odvisna od pomembnosti problema v očeh posvojitelja, ki naj bi mu ga inovacija razrešila.
	standard (ang. *standard*)	Industrija in stranke začnejo uporabljati inovacijo kot standard in zato jo je uporabnik primoran sprejeti.
	tehnološka prednost (ang. *technological edge*)	Inovacija je tehnološko naprednejša od drugih alternativ.
Naloga	**poslovna prednost** (ang. *commercial advantage*)	Možnost boljšega zadovoljevanja potreb stranke. Notranji ali zunanji prodajalec prodaja inovacijo kot koristen izdelek. Kasneje se ta izdelek komercializira.
	prepoznavanje potrebe uporabnika (ang. *user need recognition*)	Inovacija mora ustrezati potrebam uporabnika pri izvajanju njegovih nalog.
	odpor uporabnika (ang. *user resistance*)	Uporabniki nasprotujejo spremembam, ko naloge postanejo težke in/ali uporaba inovacija ni skladna z njihovimi osebnimi interesi.
Posameznik	**samostojno testiranje** (ang. *own testing*)	Inovacija se preizkuša na eksperimentalni osnovi.
	mreža osebnih stikov (ang. *personal contact network*)	Zanašanje na izkušnje ljudi iz svojih krogov.
	lastna pravila in nadzor nad lastnim delom (ang. *own rules and control of own work*)	Možnost posameznika, da eksperimentira z novo idejo in se prepriča o koristnosti inovacije.
	učenje z delom (ang. *learning by doing*)	Učenje vrednotenja inovacij na podlagi lastnih izkušenj.
Okolje	**kulturne vrednote** (ang. *cultural values*)	Naklonjenost spremembam.
	tehnološka infrastruktura (ang. *technological infrastructure*)	Razvitost tehnološke infrastrukture.
	pravila skupnosti (ang. *community norms*)	Upoštevanje pravil organizacije.

Skupina faktorjev	Faktor	Opis
	viri (ang. *funding*)	Razpoložljivost potrebnih sredstev za uporabo inovacije na predviden način.
Organizacija	**medosebne mreže** (ang. *interpersonal networks*)	Izmenjava vrednotenja inovacij med posamezniki.
	vrstniške mreže (ang. *peer networks*)	Socialne vezi.
	neformalna komunikacija (ang. *informal communication*)	Izmenjava informacij je neformalna in nenačrtovana.
	tehnološke izkušnje (ang. *technological experience*)	Tehnološke izkušnje v daljšem časovnem obdobju.
	delovne skupine (ang. *working teams*)	Člani skupine imajo primarni nadzor nad svojim upravljanjem.
	mnenjski voditelji in zastopniki sprememb (ang. *opinion leaders and change agents*)	Posameznik, ki vpliva na odločitev organizacije glede inovacije.
	soodvisnosti od drugih (ang. *interdependence from others*)	Vsak posvojitelj povečuje uporabnost inovacije za prihodnje posvojitelje (mrežne eksternalije).
	tip posvojitelja (ang. *adopter type*)	Stopnja širitve inovacije, v kateri posameznik le-to sprejme (inovator, zgodnji posnemovalec, zgodnja večina, pozna večina, zamudnik).
	hierarhija upravljanja (ang. *management hierarchy*)	Sprejetje inovacije zahteva vodstvo.

2.2.3 HIPOTEZE OVERHAGE-SCHLAUDERERJA O SPREJEMLJIVOSTI SCRUMA

Overhage in Schlauderer [14] sta raziskovala, kako razvijalci sprejemajo metodologijo Scrum. Za osnovo raziskave sta vzela Rogersov model teorije DOI, a se osredotočila le na tri faktorje iz skupine inovacija:

- relativno prednost,
- kompatibilnost,
- kompleksnost,

saj je empirična raziskava [13] pokazala, da ti trije faktorji pomembno vplivajo na stopnjo sprejemljivosti inovacije.

Postavila sta osem hipotez, kako metodologijo Scrum sprejemajo razvijalci, ki se nanašajo na prej omenjene tri faktorje iz skupine inovacija, kar prikazuje Tabela 2.

Tabela 2: Hipoteze Overhage-Schlaudererja in njihova potrditev v raziskavi

Faktor		Hipoteza	Potrditev
Relativna prednost	čas do trga	Razvijalci zaznavajo hitrejši razvoj v primerjavi s klasičnimi projekti.	da
	zahteve naročnika	Razvijalci bolje zaznavajo zahteve strank v primerjavi s klasičnimi projekti.	da
	učinki učenja	Razvijalci zaznavajo boljše učinke učenja v primerjavi s klasičnimi projekti.	da
	zadovoljstvo	Razvijalci so bolj zadovoljni z rezultati v primerjavi s klasičnimi projekti.	da
Kompatibilnost	transparentnost	Razvijalci zaznavajo večjo transparentnost poteka razvoja v primerjavi s klasičnimi projekti.	da
	sodelovanje	Razvijalci zaznavajo boljše sodelovanje v primerjavi s klasičnimi projekti.	da
Kompleksnost	kompleksnost procesa	Razvijalci zaznavajo manjšo kompleksnost procesa v primerjavi s klasičnimi projekti.	niti da niti ne
	disciplina	Razvijalci zaznavajo večjo disciplino, ki je zahtevana, v primerjavi s klasičnimi projekti.	da

Tornatzky in Klein [25] sta prišla do ugotovitve, da je razen faktorja kompleksnost zaznavanje teh faktorjev v pozitivni korelaciji s stopnjo sprejetja inovacije. Zaznavanje kompleksnosti je negativno korelirano, kar pomeni, da bolj kot je inovacija kompleksna za uporabo, manj je posameznik naklonjen njenemu sprejetju.

Overhage in Schlauderer sta podatke za raziskavo zbrala leta 2009 pri vodilni nemški zavarovalnici na svetu, kjer so Scrum začeli uvajati leta 2007. Pred tem je zavarovalnica večinoma uporabljala razširjen slapovni model, to je V-model. Scrum so izvajali, kot ga predpisuje literatura, in je zajemal predvidene vloge Scrum, srečanja in načela. V zavarovalnici so takrat razvijali 23 projektov po metodologiji Scrum, v katere je bilo vključenih okoli 200 ljudi. Intervju sta izvedla s petimi izbranimi predstavniki podjetja, ki so bili v vlogi skrbnika metodologije, trenerja Scruma ali izvršnega direktorja.

Prvih šest hipotez sta Overhage in Schlauderer s pomočjo rezultatov intervjujev potrdila in ugotovila, da pozitivno vplivajo na sprejemanje Scruma. Sedme hipoteze, ki se nanaša na kompleksnost procesa, nista niti potrdila niti ovrgla. Osmo hipotezo, ki se nanaša na disciplino, sta potrdila in zaznala, da ima negativen vpliv pri sprejemanju Scruma.

2.2.4 POSTMORTEM ANALIZA PROJEKTA

Birk, Dingsøyr in Stålhane trdijo [1], da pri vsakem projektu razvoja programske opreme člani skupine pridobijo nova znanja in izkušnje, ki lahko koristijo prihodnjim projektom in profesionalnemu razvoju posameznika. Veliko znanja ostane neopaženega in se ga ne deli med posameznike ali skupine. Njihova izkušnja s postmortem analizo projekta dokazuje, da je to odličen način za upravljanje znanja, ki zajame izkušnje in predloge za izboljšave iz zaključenih projektov. Identificirane priložnosti za izboljšave so lahko začetek trajne spremembe v podjetju.

Ob ustrezni uporabi postmortem analize člani skupine prepoznajo in se spomnijo, kaj so se naučili med projektom. Proces postmortem analize je sestavljen iz treh faz:

- priprave,
- zbiranja podatkov,
- analize.

V fazi priprave se seznanimo z zgodovino projekta, da bi bolje razumeli, kaj se je zgodilo. Pregledamo razpoložljivo gradivo in si določimo cilj postmortem analize, ki je lahko splošen, na primer izkušnje projekta, ali specifičen, na primer ocena stroškov projekta. Priporočljivo je, da se osredotočimo na obe vrsti izkušenj, in sicer tako pozitivne kot negativne.

V fazi zbiranja podatkov zberemo ustrezne izkušnje iz projekta. To lahko naredimo z različnimi tehnikami, kot so na primer delno strukturirani intervjuji, skupinska diskusija, vprašalniki in druge.

V fazi analize se prepričamo, ali smo razumeli, kar so nam vprašani povedali in ali imamo vsa pomembna dejstva.

Rezultat postmortem analize je poročilo, v katerem so povzete izkušnje projekta in izpostavljeni glavni problemi ter uspehi.

Postmortem analiza je smiselna po končanem projektu in ko podjetje išče kakovostne izkušnje, ki bi pomagale izboljšati podobne projekte v prihodnosti.

3 UVAJANJE METODOLOGIJE SCRUM NA PRAKTIČNEM PRIMERU

Od junija 2014 do novembra 2015 sem bila članica razvojne skupine na projektu prenove spletnega portala za državljane in zalednih sistemov. Izvajalec projekta, ki je bil izbran na javnem razpisu, je bilo večje slovensko IT podjetje. Programska rešitev je bila v celoti razvita po metodologiji Scrum.

3.1 POTEK PROJEKTA

Projekt je bil iz vrst prenove obstoječega informacijskega sistema. Prenova, ki je bila določena z javnim razpisom, je obsegala:

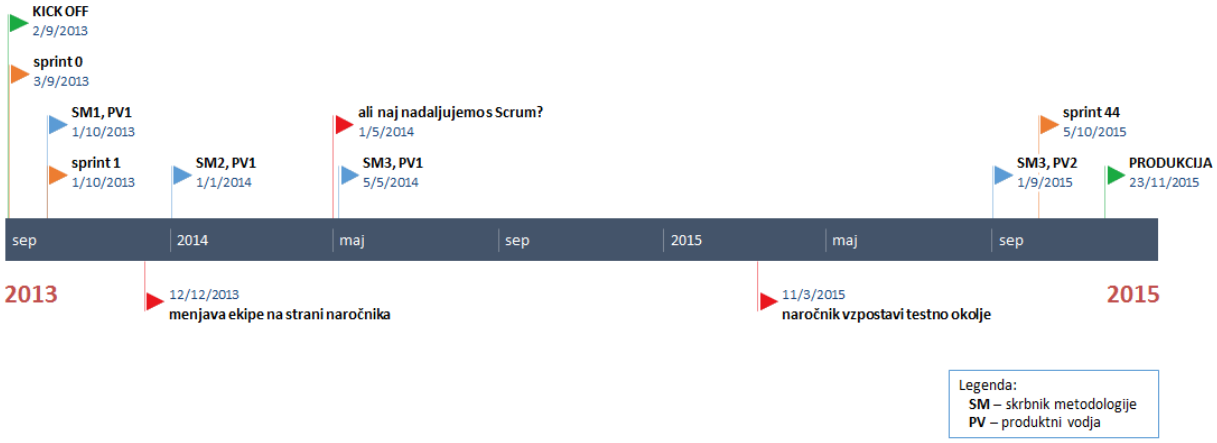
- nov design portala;
- razširitev portala z osebnimi podatki državljana;
- menjavo sistema za upravljanje vsebin (ang. *content management system*) in prilagoditev le-tega potrebam in željam naročnika;
- v zaledju je bila postavljena podatkovna baza z višjo verzijo kot na starem sistemu, posledično je bilo potrebno migrirati podatke;
- storitveni nivo je bil implementiran po SOA arhitekturi;
- za pridobivanje podatkov zunanjih virov je bilo potrebno implementirati povezave do zunanjih servisov, na primer pladenj[1], si-pass, varnostna shema, nova podpisna komponenta;
- aplikacijski strežnik iAS je bil zamenjan z JBoss, posledično je bilo treba prilagoditi vse stare sisteme za nov aplikacijski strežnik.

Ob začetku projekta se je skupina Scrum, ki je bila določena za delo na projektu, soočala z:

- za večino novo metodologijo razvoja, to je Scrum,
- novimi sodelavci, to je neuigrana skupina,
- za večino novo vsebino projekta,
- novimi tehnologijami.

[1] Pladenj je državni informacijski sistem za standardizirano izvajanje elektronskih poizvedb na podatkovne vire.

Projekt je bil obsežen in dolgotrajen. Začel se je septembra 2013 in trajal do konca novembra 2015, kar prikazuje Slika 3, trajal je torej več kot dve leti.



Slika 3: Časovnica projekta in pomembni mejniki

V času projekta je bilo veliko sprememb:

- spremembe zahtev,
- menjava sponzorja,
- menjava članov skupine,
- menjava skrbnika metodologije in
- menjava produktnega vodja.

Želja, da bo projekt izdelan po metodologiji Scrum, je prišla od nadrejenih v podjetju.

Za podjetje je bil to prvi projekt, ki je bil v celoti izveden po metodologiji Scrum. Vzporedno s tem projektom so metodologijo Scrum začeli vpeljevati še na drugem projektu, kjer se ni obnesla, zato so jo po nekaj mesecih opustili.

3.2 SKUPINA SCRUM

Skupina Scrum se je v času razvoja rešitve spreminjala; eni člani so prihajali, drugi odhajali, le 8 oseb je bilo prisotnih skozi celoten projekt. Med projektom se je razvrstilo 26 različnih oseb, ki so bili del skupine Scrum. Občasno so sodelovale še druge osebe, ki niso bile člani skupine. Na projektu je sočasno delalo 12–15 oseb. Večina oseb je prihajala iz projektov, kjer so razvijali po slapovni metodologiji, peščica jih je že imela izkušnje s Scrumom, za nekatere je bil to prvi projekt razvoja programske rešitve.

Skupina Scrum je bila ves čas projekta locirana v skupni pisarni.

3.2.1 PRODUKTNI VODJA

V celotnem projektu sta se razvrstila dva produktna vodja, kar prikazuje Slika 3.

Prvi produktni vodja, v nadaljevanju PV1, je bil zaposlen v našem podjetju in ne na strani naročnika. Imel je dolgoletne izkušnje z vodenjem projektov po tradicionalni slapovni metodologiji. Na tem projektu je prvič nastopil v vlogi produktnega vodja po Scrumu. Vloga mu je bila določena s strani nadrejenih in se nikoli ni popolnoma poistovetil z njo.

Tri mesece pred koncem projekta je njegovo vlogo prevzel drugi produktni vodja, v nadaljevanju PV2, ki je bil zunanji sodelavec izvajalca. PV2 je imel dolgoletne izkušnje s Scrumom, bil je certificiran ScrumMaster in Product Owner.

PV2 je uvedel igro škatla Kudo (ang. *Kudo Box*) [30]. Namen te igre je spodbujati pozitivni odnos med člani skupine. Posameznik, ki je opazil pozitivno dejanje ali izjemen trud sodelavca, je na kartico Kudo napisal ime in dejanje, ki ga želi pohvaliti, ter se po želji tudi podpisal. PV2 je po dnevnem sestanku Scrum pred celotno skupino prebral kartice, ki so se nabrale v škatli Kudo. Tisti, ki je bil pohvaljen, je bil deležen aplavza skupine in si je iz škatle Kudo izbral nagrado. Nagrade so bile simbolične in majhnih vrednosti, to je do 5€. Izpolnjene kartice so bile pritrjene na steni ob vhodu v pisarno, kar prikazuje Slika 4. Ta igra je prispevala k boljšemu ekipnemu duhu v skupini in posledično boljši kodi.

Slika 4: Škatla Kudo in stena Kudos

3.2.2 SKRBNIK METODOLOGIJE

Na projektu so se razvrstili trije skrbniki metodologije, kar prikazuje Slika 3.

Prvega skrbnika metodologije, v nadaljevanju SM1, so določili nadrejeni. Bil je glavni razvijalec za Javo in ni imel časa za aktivnosti Scruma. Ker se ni želel poglabljati v vsebino projekta, je težko podpiral PV1, na primer, kako razbiti vsebino na uporabniške zgodbe. Po začetnih treh mesecih ni več želel biti v tej vlogi.

Januarja 2014 je njegovo mesto prevzel drugi skrbnik metodologije, v nadaljevanju SM2, ki je ravno takrat pridobil certifikat ScrumMaster. Vpeljal je vse prakse, kot jih navaja literatura Scrum, a jih skupina Scrum ni želela izvajati, saj so se jim zdele nesmiselne. Postopoma so vse prakse ukinili in tudi SM2 ni želel biti več v tej vlogi. Ob tem času se je pojavilo tudi vprašanje, ali naj skupina s Scrumom sploh nadaljuje, saj v praksi nikakor ni zaživel tako, kot ga opisuje literatura. Vzdušje v skupini je bilo zelo slabo.

Maja 2014 je SM2 vlogo predal tretjemu skrbniku metodologije, v nadaljevanju SM3. SM3 je začel z dosledno uporabo Jire za podporo Scrumu. Postopoma je začel vpeljevati prakse Scrum in agilne igre. Ko je v proces razvoja ponovno vpeljal pregled sprinta, so tudi druge prakse počasi začele dobivati smisel, na primer, ker je skupina hotela izvajati boljše preglede sprinta, ga je morala tudi bolje načrtovati (načrtovanje sprintov), se pogovoriti o skupinskem delu (retrospektiva sprintov) in testiranju oziroma sledenju konceptu »done«. Tako so bile počasi vpeljane vse prakse z razlogom in potrebo. Vzdušje skupine se je začelo izboljševati in člani skupine Scrum so začenjali razumeti agilno miselnost. Med tem časom je SM3 pridobil certifikat ScrumMaster.

3.2.1 RAZVOJNA SKUPINA

Razvojno skupino so sestavljali načrtovalec, skrbnik baze, programerji in trije analitiki, ki so bili tudi glavni testerji.

Sprva se člani skupine niso najbolje razumeli med seboj in znotraj skupine sta nastali dve manjši skupini, v eni so bili programerji, v drugi analitiki. V komunikaciji med tema dvema skupinama je bilo veliko šuma, zato je pri implementiranju uporabniških zgodb prihajalo do težav s konceptom »done« in zelo slabega vzdušja v celotni skupini. Ker je bila klima v skupini tako slaba, so nadrejeni celotno skupino Scrum konec maja 2014 poslali na celodnevno delavnico komuniciranja. Delavnica je prispevala k boljši komunikaciji in odnosom v skupini.

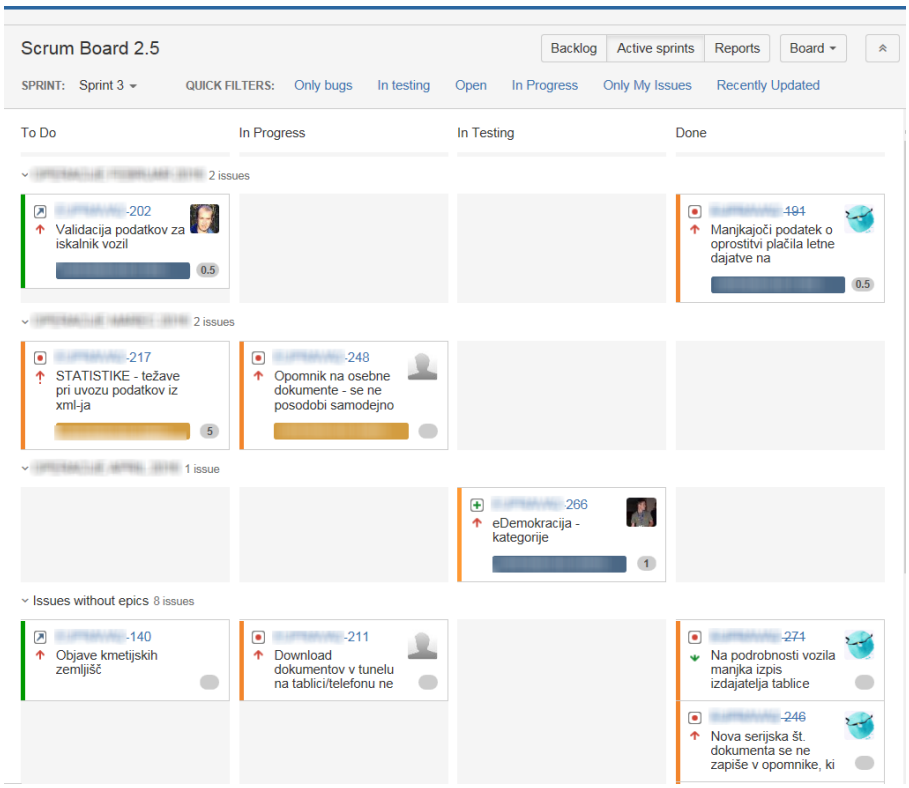
3.3 UPORABLJENE PRAKSE SCRUM

Skupina se je v času projekta učila in preizkušala prakse Scrum, saj je bil to za podjetje prvi projekt izveden po metodologiji Scrum.

3.3.1 SEZNAM ZAHTEV

Za seznam zahtev je skrbel produktni vodja. Skupaj z naročnikom se je dogovarjal za funkcionalnosti, ki jih je potrebno implementirati. Skrbel je tudi za prioritizacijo seznama zahtev.

Za upravljanje projekta je skupina uporabljala Atlassianovo Jiro, ki je po raziskavi VersionOne drugo najbolj pogosto orodje za upravljanje agilnih projektov in jo uporablja kar 51 % vprašanih [32]. S pomočjo Jire so člani skupine vzdrževali seznam zahtev izdaje in sprintov, tablo Kanban, si dodeljevali zahteve, razvijali uporabniške zgodbe in merili učinkovitost. To orodje nudi celotno podporo za delo po Scrumu in je enostavno za uporabo. Od maja 2014 je bila Jira dosledno uporabljena za podporo Scrumu, pred tem pa ne, saj je PV1 vzdrževal prioritete projekta po svojem sistemu, ker mu koncept uporabniških zgodb ni bil blizu. Slika 5 prikazuje del elektronske table Kanban enega izmed sprintov.



Slika 5: Elektronska tabla Kanban v Jiri

Na začetku projekta je bila vpeljana tudi fizična tabla Kanban, na kateri so bili lističi uporabniških zgodb in njenih nalog. Takrat ni bila usklajena z Jiro in po nekaj mesecih je bila fizična tabla Kanban opuščena zaradi neažurnosti. Slika 6 prikazuje fizično tablo Kanban, ki je maja 2015 spet zaživela in bila uporabljena do konca projekta.

Slika 6: Fizična tabla Kanban

Od maja 2015 sta bila za podporo procesu Scruma aktivna dva sistema, elektronski v Jiri in fizični v obliki table Kanban na steni v pisarni. Fizična tabla Kanban je tokrat bila usklajena z elektronsko v Jiri in je razvojni skupini ves čas omogočala celoten pogled na sprint. Pomanjkljivost elektronskega sistema je omejen prikaz elektronske table, ki je precej majhna in je potrebno pomikanje navzdol in navzgor po strani, s tem pa se izgubi celoten pregled. Ker je bil diagram Burndown uporabljen le na začetku projekta in kmalu opuščen, je tabla Kanban skupini Scrum kazala stanje sprinta, to je katere uporabniške zgodbe je potrebno razviti, katere so v razvoju, testiranju in katere so že končane. Jira je bila uporabljena predvsem za administrativno podporo, na primer vzdrževanje seznama zahtev produkta, vodenje seznamov realiziranih zgodb po sprintih, minimalno dokumentacijo, planiranje sprintov, spremljanje hitrosti skupine po sprintih v točkah uporabniških zgodb, razvojni proces uporabniških zgodb, število porabljenih ur, pa tudi za sinhronizacijo fizične table Kanban, saj samolepilni listki pogosto padajo s table. Za obe tabli je bil odgovoren skrbnik metodologije, razvijalci pa so skrbeli za ažurnost podatkov.

3.3.2 SPRINT

Prvi sprint, to je sprint 0, je trajal en mesec in je bil namenjen učenju novih tehnologij, spoznavanju vsebine in pripravi projekta za izvedbo. V tem času je bila skupina na interni predstavitvi seznanjena z metodologijo Scrum.

Nadaljnji sprinti so trajali dva tedna. Vmes je bilo nekaj sprintov zaradi praznikov ali dopustov, podaljšanih na tri tedne. Zadnji mesec pred produkcijo so bili sprinti dolgi en teden in so bili namenjeni intenzivnemu testiranju in stabilizaciji produkcijske verzije.

Naročniku so bili izdelki dostavljeni sproti po sprintih ali izdajah. Zaradi kompleksnosti projekta se je naročnik odločil, da bo izdelek v celoti dal v produkcijo šele ob koncu projekta.

3.3.3 PLANIRANJE SPRINTA, UPORABNIŠKE ZGODBE IN KONCEPT »DONE«

Uporabniške zgodbe sprva niso bile jasno definirane oziroma so bile preveč razdrobljene na naloge in nepovezane, da bi lahko videli celotno uporabniško zgodbo. Tudi ocenjevanje uporabniških zgodb na začetku projekta ni bilo učinkovito, saj se je razdrobljene naloge ocenjevalo v urah. Primer nalog uporabniške zgodbe: načrtovanje designa, implementacija designa, implementacija na bazi, implementacija v CMS sistemu, dokumentiranje in testiranje. Ob koncu sprinta so bile vse planirane zgodbe delno implementirane, saj se je med temi razdrobljenimi nalogami izgubila celotna slika posamezne uporabniške zgodbe. Tako načrtovanje sprinta ni imelo smisla in je bilo kmalu opuščeno.

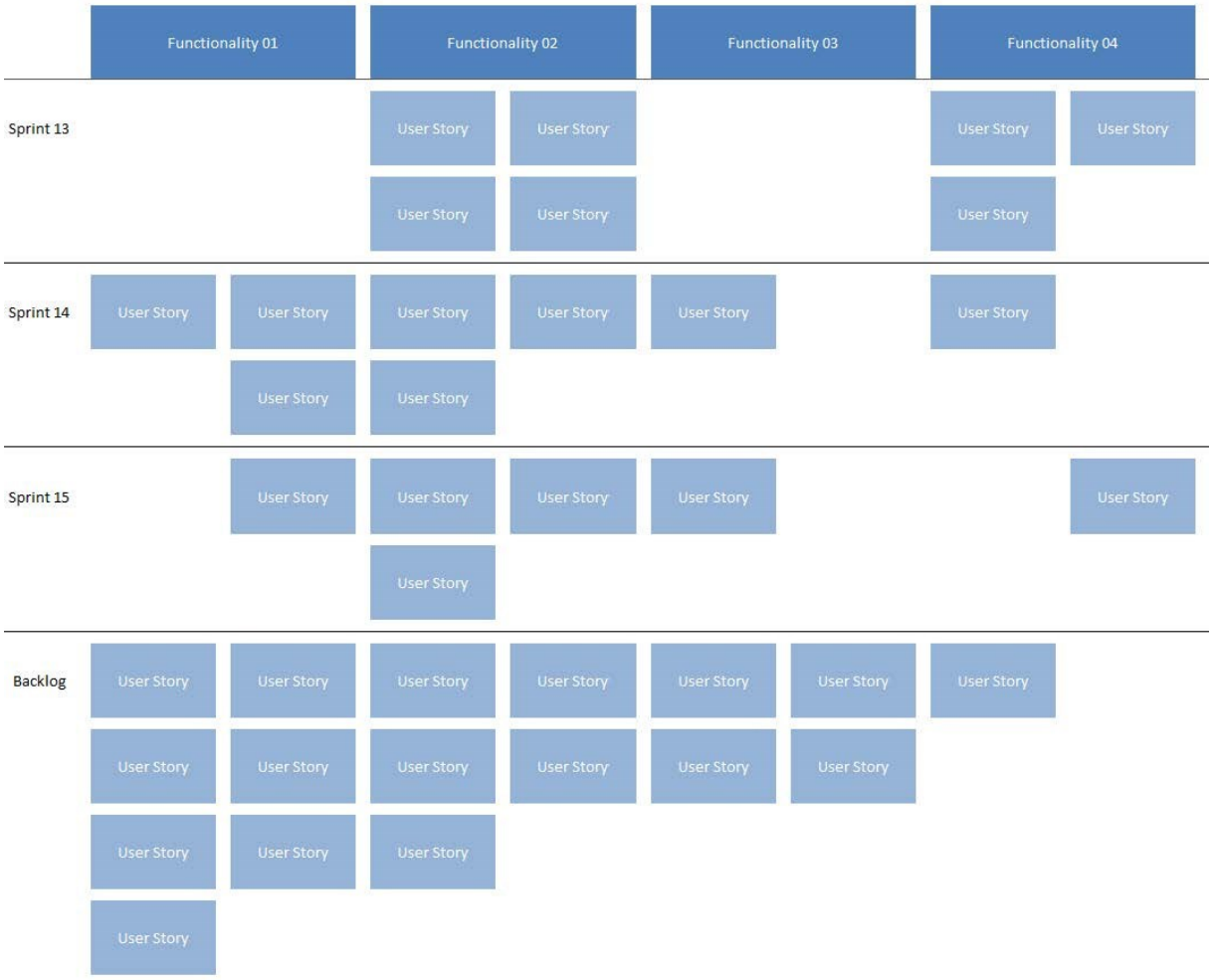
SM3 je januarja 2015 uvedel ocenjevanje posamezne uporabniške zgodbe kot celote in za vsako uporabniško zgodbo je skupina definirala naloge. Uporabniške zgodbe so člani skupine Scrum ocenjevali v urah. V Jiri je bila dosledno kreirana ena naloga Jira (ang. *Jira task*) za vsako uporabniško zgodbo. Nalogo Jira so si člani skupine podajali med seboj in tako je bil dosežen pregled nad posamezno uporabniško zgodbo kot celoto. S tem je skupina začela bolj dosledno izvajati koncept »done«, na kar je skupino po potrebi opozarjal SM3. Ob koncih sprinta so bile izdelane delujoče uporabniške zgodbe. S takim načinom dela se PV1 sprva ni strinjal, a ga je SM3 prepričal v smiselnost uporabe, saj se je število delujočih zgodb ob koncu sprinta močno povečalo.

Od januarja 2015 je planiranje potekalo vsako drugo sredo od 9:00 do 11:00, prej pa po dogovoru v skupini.

Po mnenju SM3 ocenjevanje v urah ni bilo učinkovito, ker naj bi razvijalci planirali preveč pesimistično, zato je junija 2015 uvedel ocenjevanje v točkah (ang. *story points*) s pomočjo igre *Planning Poker* [29].

Proti koncu projekta je bila skupina seznanjena z grupiranjem uporabniških zgodb po funkcionalnostih in sprintih (ang. *User Story Mapping*) [16]. Koncept grupiranja uporabniških zgodb prikazuje Slika 7. Tako grupiranje uporabniških zgodb kaže sliko še neizdelanih

funkcionalnostih in pomaga pri vzdrževanju prioritet seznama zahtev izdelka. Funkcionalnost (ang. *epic*) je na projektu predstavljala skupina uporabniških zgodb.



Slika 7: Grupiranje uporabniških zgodb po funkcionalnostih in sprintih

3.3.4 DNEVNI SESTANEK IN DIAGRAM BURNDOWN

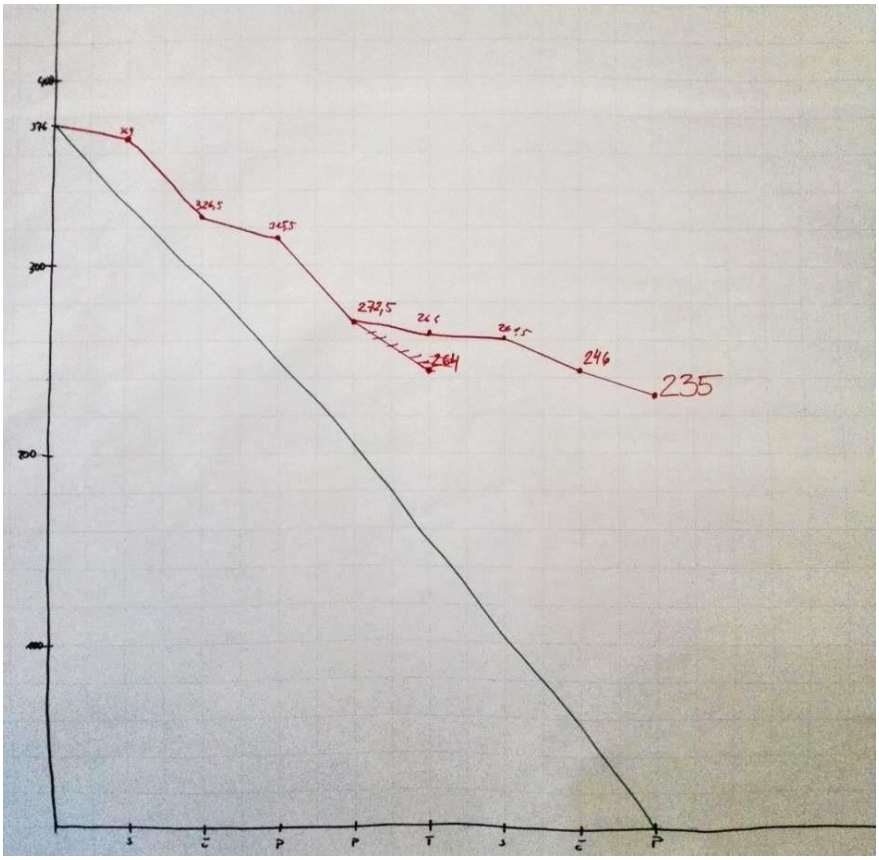
Dnevni sestanek (ang. *Daily Scrum*) je potekal vsak dan od 8:40 do 9:00, razen na dan planiranja sprinta. Na njem je vsak član skupine odgovoril na sledeča vprašaja:

- Kaj **je** bilo doseženo od zadnjega sestanka?
- Kaj **bo** narejeno do naslednjega sestanka?
- Katere **ovire** so v napoto?

Za časovno omejitev in fokus sestanka je skrbel skrbnik metodologije.

Oseba, ki je bila kasneje SM2, je v času SM1 uvedla dnevno merjenje učinkovitosti skupine z diagrami Burndown v urah. Podatki so se zbirali tako, da je vsak član skupine na dnevnem

sestanku na samolepilni listek naloge napisal, koliko ur še potrebuje za izvedbo naloge, na kateri dela. Bodoči SM2 je po dnevnem sestanku seštel vpisane ure in jih vrisal na papir, kjer je vzdrževal diagram Burndown, kar prikazuje Slika 8. Slabi rezultati diagrama so skupini jemali motivacijo, zato člani skupine niso dosledno vnašali ur za dokončanje nalog uporabniških zgodb in diagram Burndown je bil kmalu opuščen.



Slika 8: Diagram Burndown za nadzor dela 2. sprinta

3.3.5 PREGLED SPRINTA

Pregled sprinta je sprva vodil PV1, na katerem je predstavil izdelke sprinta oziroma scenarije delavnic, ki jih je skupina razvijalcev pripravljala za naročnika vzporedno z razvojem rešitve. Pregledi sprinta sprva niso bili redni, saj je imela razvojna skupina v začetku težave s konceptom »done«, zato izdelki ob koncu sprinta pogosto niso bili delujoči.

SM3 je septembra 2014 odločil, da bodo razvijalci sami predstavljali izdelke na pregledih sprinta. Zamislil si je igro *Best Sprint Task Force*, kar prikazuje Slika 9. To je igra, kjer ob koncu sprinta SM razglasi skupino, ki je najbolje izvedla uporabniško zgodbo. Zmagovalno skupino je upodobil s karikaturo in jo obesil na vidno mesto v pisarni. Na preglede sprinta je začel prinašati piškote. Vse te vpeljave so povzročile pozitivne spremembe:

- več izdelanih in delujočih uporabniških zgodb sprinta,
- zmanjšan odpor razvijalcev do pregledov sprinta,
- pregledi sprinta so postali sproščeni.

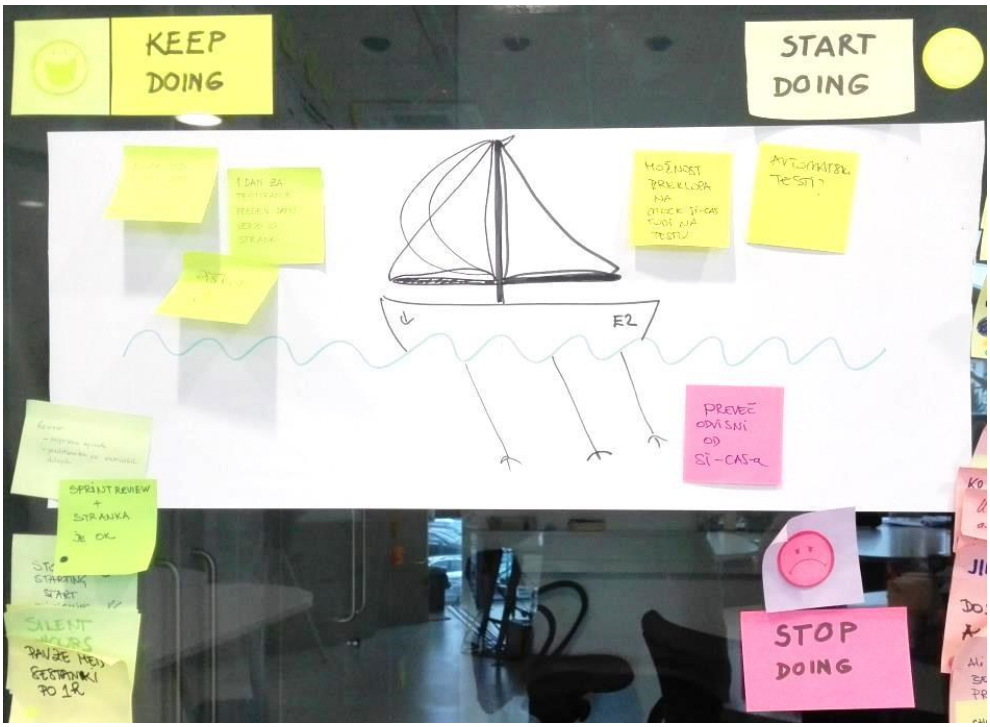
Slika 9: Igra *Best Sprint Task Force*

3.3.6 RETROSPEKTIVA SPRINTA

Retrospektiva je zelo pomemben sestanek, saj je povratna informacija nujna za rast skupine in dolgoročno sprejemljivost Scruma [28].

Retrospektive na projektu nikoli niso bile stalnica ob koncih sprinta. Ko je SM3 februarja 2015 uvedel igro jadrnica (ang. *sailboat*), kar prikazuje Slika 10, so retrospektive sprintov postale bolj priljubljene. Člani skupine so lažje izrazili:

- stvari, ki pomagajo skupini, da gre lažje čez sprint in bi jih obdržali – **veter v jadrih**,
- stvari, ki skupino upočasnjujejo in jih je treba odstraniti – **sidra**,
- stvari, ki bi jih uvedli, da bi bili kot skupina še boljši.



Slika 10: Retrospektiva sprinta – igra jadrnica

Skupina je na vsaki retrospektivi sprinta izmed vseh predlogov skupaj izbrala tiste, ki jih bo izvajala v naslednjem sprintu. Na naslednji retrospektivi sprinta je skupina na jadrnici preverila, ali je izbrane predloge prejšnjega sprinta upoštevala. Če jih ni, se je ponovno pogovorila o razlogih za neupoštevanje.

Retrospektive sprinta so trajale pol ure in bile izvedene takoj po pregledu sprinta.

4 NAČRT ŠTUDIJE

Študija mora odgovoriti na v uvodu zastavljene štiri raziskovalne cilje, zato smo študijo načrtovali tako, da smo predvideli zbiranje preliminarnih rezultatov s pomočjo vprašalnika.

Obsežen vprašalnik, ki so ga izpolnjevali člani skupine Scrum, smo razdelili na pet sklopov. Prvi sklop je namenjen splošnim informacijam anketiranca. V drugem sklopu se vprašanja nanašajo na stopnjo sprejetja posameznih praks Scrum, kot jih navajajo Kwon, Cooper in Zmud in so povezani z RC1. Tretji sklop obsega trditve o faktorjih, ki po Rogersu vplivajo na dolgoročno sprejemljivost inovacije in so povezani z RC2. V četrtem sklopu se vprašanja nanašajo na osem hipotez, ki jih predpostavljata Overhage in Schlauderer, kako razvijalci sprejemajo faktorje Scruma in so povezani z RC3. V zadnjem, to je petem sklopu, so specifična vprašanja na podlagi značilnosti projekta prenove spletnega portala in zalednih sistemov, ki so povezani z RC4.

Dodatno smo želeli osvetliti in potrditi rezultate vprašalnika, zato smo načrtovali delno strukturirane intervjuje s tipičnimi predstavniki skupine Scrum. Vsebino delno strukturiranega intervjuja smo oblikovali naknadno, to je po analizi rezultatov vprašalnika in je predstavljena v poglavju 6.

4.1 SPLOŠNA VPRAŠANJA O ANKETIRANCU

V prvem delu vprašalnika so štiri splošna vprašanja o anketirancu. Sprašujemo ga o:

- vlogi, v kateri je nastopal na projektu,
- izkušnjah s Scrumom (leta),
- skrbnikih metodologije, ki jih je imel,
- produktnih vodjih, ki jih je imel.

Anketiranec je lahko izbiral med ponujenimi odgovori. Na prvi dve vprašanji je bil možen samo en odgovor, na drugi dve več odgovorov.

Podrobnosti vprašalnika so v prilogi Dodatek A.

4.2 STOPNJE SPREJETJA POSAMEZNIH PRAKS SCRUM

Drugi del vprašalnika se nanaša na oceno usposobljenosti posameznika za izvajanje 15 tipičnih praks Scruma po končanem projektu. 15 tipičnih praks prikazuje Tabela 3.

Tabela 3: Prakse Scrum, ki so v drugem delu vprašalnika

Praksa Scrum	Opis
Vzdrževanje seznama zahtev izdelka (ang. *product backlog*)	Seznam zahtev je prioritiziran seznam zahtev, ki so potrebne za razvoj produkta. Produktni vodja je odgovoren za njegovo vsebino, prioritizacijo in dostopnost. Seznam ni nikoli dokončen, saj se ves čas projekta stalno dopolnjuje.
Uporaba uporabniških zgodb (ang. *user stories*) za specifikacijo zahtev	Vsaka zahteva iz seznama zahtev je predstavljena kot uporabniška zgodba, ki sestoji iz kratkega opisa in testnih scenarijev za sprejem izdelane zgodbe. Zgodba je v jeziku stranke ročno napisana na lističu.
Sodelovanje s produktnim vodjem (ang. *product owner*)	Podrobnosti, povezanih z realizacijo posameznih uporabniških zgodb, ne dokumentiramo, ampak jih razčiščujemo v pogovorih s produktnim vodjem.
Ocena napora za uporabniško zgodbo	Za potrebe planiranja se uporabniške zgodbe ocenjuje v točkah s pomočjo tehnike *Planning Poker*.
Ocena napora za uporabniško zgodbo	Za potrebe planiranja se velikost uporabniške zgodbe ocenjuje s pomočjo tehnike *Team Estimation Game*.
Načrtovanje izdaje (ang. *release planning*) na osnovi ocenjene hitrosti (ang. *velocity*) razvojne skupine	Glede na ocenjeno hitrost skupine se uporabniške zgodbe po prioritetah porazdeli v sprinte, da se določi okviren datum zaključka.
Načrtovanje sprinta (ang. *sprint planning*)	Na začetku vsakega sprinta se produktni vodja sestane s skupino, da opredelijo vsebino naslednjega sprinta. Količina dogovorjenega dela ne sme preseči ocenjene hitrosti skupine.
Vzdrževanje seznama zahtev sprinta (ang. *sprint backlog*)	V drugem delu načrtovanja sprinta skupina pripravi seznam nalog, s katerimi izbrane uporabniške zgodbe spremeni v potencialno delujoče funkcionalnosti. Naloge se ocenijo in dodelijo članom skupine. Med samim sprintom se po potrebi dodaja nove naloge in prevelike naloge razbija na manjše.

Praksa Scrum	Opis
Dnevni sestanki (ang. *Daily Scrum*)	Skupina se vsak dan sestane za največ 15 min, kjer vsak član odgovarja na tri vprašanja: »Kaj si naredil od prejšnjega sestanka?«, »Kaj nameravaš narediti do naslednjega sestanka?« in »Ali imaš kakšne ovire na poti?«.
Spremljanje napredka z diagramom Burndown	Diagram Burndown prikazuje preostanek količine dela posameznega sprinta ali izdaje po času, to je vizualizacija korelacije dela, ki ga je še potrebno izvesti do konca sprinta ali izdaje in napredka skupine pri zmanjšanju tega dela.
Dosledno upoštevanje koncepta »done«	Produktni vodja sprejema le tiste zgodbe, ki so v skladu s konceptom »done«. Koncept »done« zagotavlja, da je uporabniška zgodba razvita in testirana, kar pomeni, da ne vsebuje napak in ni potrebno nobenega dela več, da bi bila dokončana.
Pregled sprinta (ang. *sprint review*)	Ob zaključku sprinta skupina predstavi rezultate sprinta produktnemu vodju in drugim zainteresiranim.
Retrospektiva sprinta (ang. *sprint retrospective*)	Pred novim sprintom skrbnik metodologije in razvojna skupina pregledajo razvojni proces preteklega sprinta in podajo predloge za izboljšave v naslednjem sprintu.
Poznavanje aktivnosti posamezne vloge Scruma (produktni vodja, skrbnik metodologije in razvojna skupina)	Produktni vodja skrbi za vizijo izdelka, vzdržuje seznam zahtev, odgovarja na vsebinska vprašanja in ocenjuje izdelane uporabniške zgodbe. Skrbnik metodologije skrbi, da vsi sledijo pravilom in praksam metodologije Scrum. Razvojna skupina deluje kot samoorganizirana, samoupravna in navzkrižno delujoča skupina, ki je kolektivno odgovorna za uspeh projekta.
Grupiranje uporabniških zgodb po funkcionalnostih in sprintih (ang. *User Story Mapping*)	Grupiranje uporabniških zgodb omogoča organiziranje uporabniških zgodb po funkcionalnostih in sprintih, kar daje celotno sliko izdelka in pomaga razumeti delovanje sistema, opredeliti pomanjkljivosti in učinkovito načrtovanje celovitih izdaj, ki prinašajo vrednost za naročnika z vsako izdajo.

Za izbrano prakso je bilo na voljo šest odgovorov, to je od »a« do »f«, ki ustrezajo šestim stopnjam sprejemanja inovacije po Kwonu, Zmudu in Cooperju [4, 9], kar prikazuje Tabela 4.

Tabela 4: Možni odgovori na vprašanje o stopnji sprejetja posamezne prakse

Možni odgovori		Nivo sprejetja
a	Za [prakso] slišim prvič.	iniciacija
b	Nisem še izvajal [prakse].	posvojitev
c	Imam dovolj potrebnih znanj, da lahko izvajam [prakso].	prilagoditev
d	S [prakso] imam že pozitivne izkušnje.	uporaba
e	[Prakso] sem izvajal že tolikokrat, da je to zame postalo rutinsko opravilo.	rutinizacija
f	[Prakso] izvajam rutinsko, učinkovito in celovito, kar olajša izvedbo ostalih ključnih aktivnosti Scruma in pohitri ter izboljša kakovost razvoja programske rešitve.	infuzija

Vsebina drugega dela vprašalnika je podrobneje predstavljena v prilogi Dodatek B.

4.3 FAKTORJI, KI VPLIVAJO NA SPREJEMLJIVOST SCRUMA

Tretji del vprašalnika sestavljajo trditve, ki se nanašajo na Rogersovo teorijo difuzije inovacije. Trditve so razdeljene na pet skupin faktorjev DOI, ki jih predstavlja Tabela 5.

Tabela 5: Faktorji DOI, ki so v tretjem delu vprašalnika

Faktorji DOI	Faktor	Trditev
Inovacija	relativna prednost	Scrum je za razvoj programske opreme boljši kot tradicionalen način.
	preprostost uporabe	Scrum je lažje razumeti in uporabljati v praksi.
	kompatibilnost	Scrum se bolj ujema z mojim želenim načinom dela, vrednotami in izkušnjami.
	možnost opazovanja, kako inovacijo uporabljajo drugi	Možno je opazovati drugo skupino pri uporabi Scruma.
	možnost preizkušanja	Scrum je mogoče preizkusiti v delovnem okolju.

Faktorji DOI	Faktor	Trditev
	strošek	Uporaba Scruma ne zahteva dodatnih stroškov in napora.
	razreševanje problema	Scrum rešuje veliko težav v zvezi z razvojem programske opreme.
	standard	Scrum je postala standardna metodologija za razvoj programske opreme.
	tehnološka prednost	Scrum je naprednejši od drugih metodologij za razvoj programske opreme.
Naloga	poslovna prednost	Scrum pomembno prispeva k boljšemu zadovoljstvu strank.
	prepoznavanje potrebe uporabnika	Scrum ustreza potrebam razvijalcev pri razvoju programske opreme.
	odpor uporabnika	Scrum poenostavlja izvajanje zahtevnih nalog.
Posameznik	samostojno testiranje	S Scrumom lahko samostojno eksperimentiram.
	mreža osebnih stikov	Prijatelji in kolegi zelo priporočajo uporabo Scruma.
	lastna pravila in nadzor nad lastnim delom	Scrum je mogoče enostavno prilagoditi mojemu načinu dela.
	učenje z delom	Scrum se lahko naučim med praktičnim delom na projektu.
Okolje	kulturne vrednote	Scrum je skladen z vrednotami okolja, v katerem delam.
	tehnološka infrastruktura	Imamo celotno tehnološko infrastrukturo, ki je potrebna za uspešno uporabo Scruma.
	pravila skupnosti	Scrum uporabljamo točno tako, kot je predpisan v literaturi.
	viri	Za uporabo Scruma imamo na razpolago vse potrebne vire, tj. literaturo, izobraževanje, čas za delo, dostop do trenerja Scrum.
Organizacija	medosebne mreže	Večina sodelavcev močno priporoča uporabo Scruma.
	vrstniške mreže	Programsko rešitev razvijam v skupini s sodelavci, s katerimi se tudi sicer veliko družim.
	neformalna komunikacija	Scrum spodbuja spontano in neformalno komunikacijo med člani skupine.
	tehnološke izkušnje	Z metodologijami razvoja programske opreme imam veliko izkušenj.

Faktorji DOI	Faktor	Trditev
	delovne skupine	Scrum omogoča, da sami načrtujemo delo in razrešimo večino problemov in težav, do katerih pride med razvojem programske opreme.
	mnenjski voditelji in zastopniki sprememb	Večina oseb, katerih mnenje spoštujem, zelo priporoča uporabo Scruma.
	soodvisnosti od drugih	Vsak nov uporabnik Scruma bistveno poveča uporabnost Scruma.
	tip posvojitelja	Nove tehnologije navadno začnem uporabljati prej kot moji kolegi in prijatelji.
	hierarhija upravljanja	Zahteva podjetja je glavni razlog, da uporabljam Scrum.

Vsaka skupina faktorjev vsebuje od tri do devet faktorjev. Trditve vprašalnika so po skupinah prikazane v prilogi Dodatek C.

Za vsako trditev je anketiranec izbral stopnjo strinjanja s pomočjo Likertove 7-stopenjske lestvice: 1 – »sploh se ne strinjam«, 2 – »se ne strinjam«, 3 – »delno se ne strinjam«, 4 – »niti se ne strinjam niti se strinjam«, 5 – »delno se strinjam«, 6 – »strinjam se«, 7 – »popolnoma se strinjam«.

4.4 VPRAŠANJA ZA PRIMERJAVO S HIPOTEZAMI OVERHAGE-SCHLAUDERER

V četrtem delu vprašalnika so trditve, ki se nanašajo na hipoteze Overhage in Schlaudererja [14] glede dolgoročne sprejemljivosti Scruma. Raziskovala sta sprejemljivost treh faktorjev iz skupine inovacija, ki naj bi najbolj vplivali na stopnjo sprejetja, in sicer: relativne prednosti, kompatibilnosti in kompleksnosti.

Trditve vprašalnika, ki se nanašajo na vsako od omenjenih hipotez, so podrobneje predstavljene v prilogi Dodatek D.

Za vsako trditev je anketiranec izbral stopnjo strinjanja s pomočjo Likertove 7-stopenjske lestvice: 1 – »sploh se ne strinjam«, 2 – »se ne strinjam«, 3 – »delno se ne strinjam«, 4 – »niti se ne strinjam niti se strinjam«, 5 – »delno se strinjam«, 6 – »strinjam se«, 7 – »popolnoma se strinjam«.

4.5 SPECIFIČNA VPRAŠANJA NA PODLAGI ZNAČILNOSTI PROJEKTA

V petem delu vprašalnika so trditve, ki se nanašajo na projekt, to je prenova portala za državljane in zalednih sistemov. Trditve smo osnovali na naših zaznavanjih pozitivnih in negativnih izkušenj z metodologijo Scrum na samem projektu in ne izhajajo iz strokovne literature, ampak so specifične za obravnavani projekt.

Glede na začetne težave vpeljave metodologije Scrum smo želeli preveriti, kaj je bilo tisto, kar je obrnilo izvajanje Scruma na bolje in omogočilo uspešno zaključen projekt. Želeli smo izvedeti, kako se je skupina razvijala v času projekta, kje so naleteli na ovire in kako so jih premagali. Zaradi premajhne vključenosti stranke nas je zanimalo, ali so anketiranci to občutili in če so, kje se je to kazalo. Zanimalo nas je, ali so vse te igre, ki so bile vpeljane v drugi polovici projekta, imele kakšen učinek na vzdušje v skupini in pri izboljšanju uporabe posameznih praks. Ne nazadnje nas je zanimalo, ali si vprašani tudi v prihodnje želijo razvijati po metodologiji Scrum, kar je pokazatelj, da je Scrum boljši kot tradicionalne metodologije.

Menimo, da so tudi te trditve vredne obravnave, saj lahko doprinesejo k boljši postmortem analizi dela na projektu in posledično boljši uporabi Scruma pri nadaljnjem delu.

Trditve so podrobneje predstavljene v prilogi Dodatek E.

Za vsako trditev je anketiranec izbral stopnjo strinjanja s pomočjo Likertove 7-stopenjske lestvice: 1 – »sploh se ne strinjam«, 2 – »se ne strinjam«, 3 – »delno se ne strinjam«, 4 – »niti se ne strinjam niti se strinjam«, 5 – »delno se strinjam«, 6 – »strinjam se«, 7 – »popolnoma se strinjam«.

5 REZULTATI IN ANALIZA VPRAŠALNIKOV

Vprašalnik, katerega strukturo predstavlja poglavje 4, je izpolnilo 15 oseb, ki so sodelovale na obravnavanem projektu. Od tega jih je sedem sodelovalo v celotnem projektu, tri osebe so prišle po začetku projekta, ena je odšla tik pred koncem projekta, štiri so sodelovale le nekaj mesecev med projektom, to je 2–4 mesecev.

Osebe so izpolnjevale vprašalnik po končanem projektu, zato rezultati odražajo stanje ob koncu projekta.

Rezultati prvega dela ankete so prikazani z grafom, za ostale dele, to je od dva do pet, so prikazani s povprečjem, mediano in standardnim odklonom.

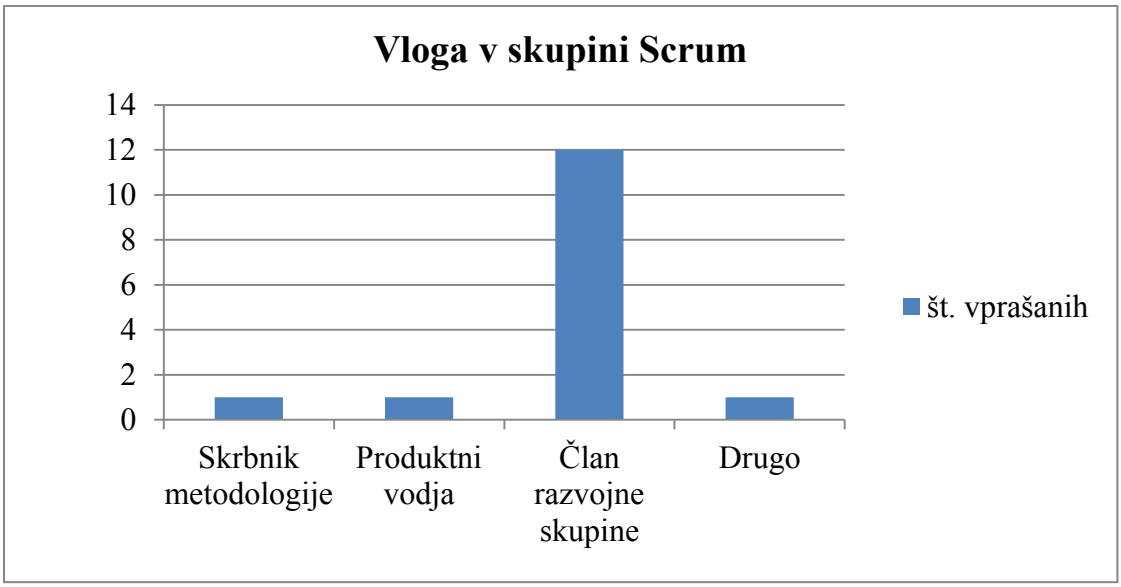
5.1 REZULTATI SPLOŠNIH VPRAŠANJ

Splošna vprašanja se nanašajo na vlogo, ki jo je posameznik imel v skupini Scrum in izkušnje z uporabo metodologije Scrum.

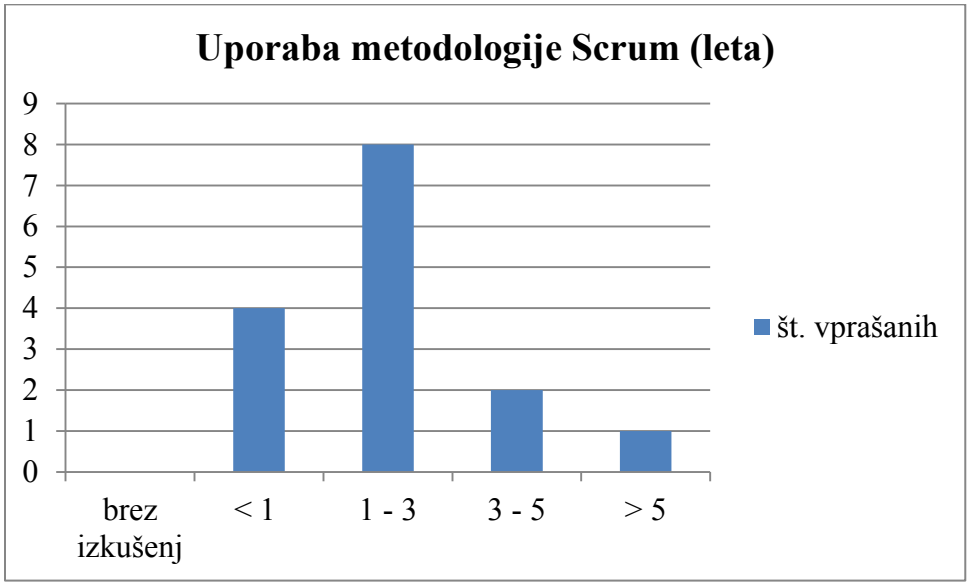
Izmed vseh vprašanih je ena oseba nastopala v vlogi skrbnika metodologije, ena v vlogi produktnega vodja, dvanajst kot člani razvojne skupine in ena v nobeni od vlog Scruma, kar prikazuje Slika 11.

Slika 12 prikazuje čas uporabe metodologije Scrum v letih. Podatki kažejo, da vsi vprašani že imajo izkušnje z uporabo metodologije Scrum. Več kot polovica vprašanih oseb, to je osem, ima izkušnje z uporabo metodologije Scrum od enega do treh let. Štirje uporabljajo metodologijo Scrum manj kot eno leto, dve osebi od tri do pet let in ena več kot pet let. Uporabnika brez izkušenj z metodologijo Scrum med vprašanimi ni bilo.

Za osebe, ki metodologijo uporabljajo manj kot 3 leta, lahko rečemo, da so začeli uporabljati Scrum prav na tem projektu, saj je projekt trajal več kot dve leti.



Slika 11: Struktura vlog Scruma vprašanih



Slika 12: Izkušenost pri uporabi Scruma

5.2 REZULTATI STOPENJ SPREJETJA POSAMEZNIH PRAKS SCRUM

Tabela 6 prikazuje zbrane rezultate, ki se nanašajo na stopnje sprejetja praks Scrum in odgovarjajo na raziskovalni cilj RC1.

Tabela 6: Stopnje sprejetja praks Scrum

Praksa Scrum	Povprečje	Mediana	Standardni odklon
Seznam zahtev	3,40	3	1,18
Uporabniške zgodbe	3,93	4	1,10
Produktni vodja	4,13	4	0,99
Igra *Planning Poker*	5,00	5	1,07
Igra *Team Estimation Game*	2,07	2	1,22
Načrtovanje izdaje	3,20	4	1,08
Načrtovanje sprinta	3,60	4	1,12
Seznam zahtev sprinta	3,47	3	1,19
Dnevni sestanki Scrum	5,47	6	1,06
Diagram Burndown za nadzor poteka dela	2,60	2	1,06
Koncept »done«	4,67	4	1,11
Pregled sprinta	5,00	5	1,07
Retrospektiva sprinta	4,40	5	1,30
Poznavanje aktivnosti posamezne vloge Scruma	5,40	6	0,74
Grupiranje uporabniških zgodb po funkcionalnostih in sprintih	3,33	4	1,23

Glede na rezultate in zastavljen raziskovalni cilj RC1 lahko rečemo, da je skupina dve praksi usvojila do stopnje »posvojitev« (mediana 2), dve do stopnje »prilagoditev« (mediana 3), šest praks do stopnje »uporaba« (mediana 4). Za tri prakse se je izkazalo, da so dosegle stopnjo »rutinizacija« (mediana 5) in dve sta dosegli stopnjo »infuzija« (mediana 6).

Najbolj sprejeti praksi sta dnevni sestanki Scrum in poznavanje aktivnosti skupine Scrum, medtem ko sta najmanj sprejeti praksi ocenjevanje uporabniških zgodb po metodi *Team Estimation Game* in uporaba diagrama Burndown za nadzor poteka dela.

Dnevni sestanek Scrum je tudi po raziskavi VersionOne daleč najbolj razširjena praksa [32], kar je razumljivo, saj jo skupina Scrum izvaja zelo pogosto, to je vsak dan, kar zagotovo prispeva k višji stopnji sprejetja prakse. Tudi poznavanje aktivnosti skupine Scrum je dobro sprejeta praksa in menimo, da iz istega razloga kot dnevni sestanek, saj se mora skupina

Scrum na dnevnem nivoju pogovarjati o posameznih vlogah in njihovih aktivnostih, da vsak član ve, kaj se od njega pričakuje. Prišli smo do podobnih ugotovitev kot Mahnič in Hovelja [11], ki sta ugotovila, da so prakse, ki se izvajajo vsak dan, bolje sprejete kot tiste, ki so periodične.

Slabše sprejetje prakse *Team Estimation Game* ni kritično, saj je skupina za ocenjevanje uporabniških zgodb uporabljala alternativno, to je igra *Planning Poker*, ki pa je bila dobro sprejeta.

Tudi pregled in retrospektiva sprinta sta dobro sprejeti praksi, kar pomeni, da so bile navkljub začetnim težavam spremembe, ki jih je vpeljal skrbnik metodologije SM3, smiselne in učinkovite ter so prispevale k večji sprejetosti teh dveh praks.

Slabša sprejetost prakse diagram Burndown je kritična, saj je spremljanje napredka razvoja zelo pomembno. Rezultat kaže na premajhno poudarjanje spremljanja razvoja ali na nezadostno usposobljenost skrbnika metodologije. Tej praksi bi morali v prihodnje posvetiti več pozornosti.

Če povprečno oceno posamezne prakse zaokrožimo navzgor, lahko rečemo, da je skupina za 10 praks dosegla stanje po usvojitvi uporabe inovacije [20], le za pet od petnajstih praks je skupina dosegla stanje pred usvojitvijo. Če pa zanemarimo najslabše sprejeto prakso, to je *Team Estimation Game*, ki je alternativa *Planning Poker*ju, lahko rečemo, da so bile prakse sprejete med 3. in 6. stopnjo, kar pomeni, da nekatere prakse poznajo in jih lahko začnejo uporabljati v praksi, z nekaterimi imajo že pozitivne praktične izkušnje, nekaj jih izvajajo že rutinsko, z dnevnim sestankom Scrum, ki je edini dosegel 6. stopnjo, pa si pohitrijo delo.

5.3 REZULTATI FAKTORJEV, KI VPLIVAJO NA SPREJEMLJIVOST SCRUM

Faktorji, ki vplivajo na sprejemljivost Scruma, so razdeljeni v pet skupin:

- inovacija,
- naloga,
- posameznik,
- okolje,
- organizacija.

V nadaljevanju so v tem vrstnem redu predstavljeni rezultati vprašalnikov za posamezno skupino faktorjev in so povezani z raziskovalnim ciljem RC2.

5.3.1 VPLIV FAKTORJEV IZ SKUPINE INOVACIJA

Rezultate, ki se nanašajo na stopnjo difuzije faktorjev inovacija, prikazuje Tabela 7.

Tabela 7: Vpliv skupine faktorjev inovacija na stopnjo difuzije

Faktor	Povprečje	Mediana	Standardni odklon
Relativna prednost	6,27	6	0,80
Preprostost uporabe	5,27	6	1,49
Kompatibilnost	6,33	7	0,90
Možnost opazovanja, kako inovacijo uporabljajo drugi	3,53	4	1,85
Možnost preizkušanja	4,13	4	1,55
Strošek	3,67	4	1,59
Razreševanje problema	5,53	6	1,25
Standard	4,93	5	1,39
Tehnološka prednost	5,93	6	1,03

Rezultati kažejo, da se noben izmed faktorjev skupine inovacija ne zdi nepomemben, saj za nobenega mediana ni 3 ali manj. Vprašani smatrajo kot »niti pomembne niti nepomembne« tri faktorje (mediana 4), enega kot »nekoliko pomembnega« (mediana 5), štiri faktorje kot »pomembne« (mediana 6) in enega kot »zelo pomembnega« (mediana 7).

Glede na raziskovalni cilj RC2, ti rezultati kažejo, da so faktorji iz skupine inovacija, ki imajo največji vpliv na pripravljenost anketirancev za sprejem Scruma kot nove metodologije, mediana 6 in več, naslednji: kompatibilnost, relativna prednost, preprostost uporabe, razreševanje problema in tehnološka prednost.

Z drugimi besedami, naši rezultati kažejo, da je Scrum pomembna izboljšava v primerjavi s tradicionalnimi metodologijami razvoja programske opreme, saj se bolj ujema z željami posameznikov, je boljša in bolj napredna kot druge metodologije, čeprav posamezniki niso imeli veliko možnosti, da bi opazovali druge skupine pri uporabi Scruma. Ti rezultati kažejo, da se anketiranci zavedajo, da se način dela spreminja in da je v današnjem času veliko sprememb, ki se jim je potrebno hitro prilagoditi, kar Scrum omogoča. Prilagajanje spremembam je šibka točka tradicionalnih metodologij, kjer spremembe med razvojem niso dobrodošle. Prav hitrost pri prilagajanju spremembam ohranja konkurenčnost na trgu tako naročnika kot tudi izvajalca.

Visoka ocena faktorja preprostost uporabe kaže na to, da je Scrum preprost za uporabo, kar pozitivno vpliva na njegovo sprejemljivost.

Odpor do uporabe Scruma lahko nastane zaradi dodatnih stroškov in napora, ki so potrebni za njegovo uporabo, če so seveda večji, kot si jih uporabnik Scruma lahko privošči.

5.3.2 VPLIV FAKTORJEV IZ SKUPINE NALOGA

Tabela 8 prikazuje rezultate vprašalnika, ki se nanašajo na stopnjo difuzije faktorjev naloge.

Tabela 8: Vpliv skupine faktorjev naloga na stopnjo difuzije

Faktor	Povprečje	Mediana	Standardni odklon
Poslovna prednost	5,67	6	0,82
Prepoznavanje potrebe uporabnika	5,73	6	0,96
Odpor uporabnika	6,07	6	1,03

Rezultati vseh treh faktorjev skupine naloga kažejo, da so ti »pomembni« pri sprejemanju Scruma, saj je pri vseh treh mediana 6.

Ti rezultati nakazujejo, da imajo vsi faktorji skupine naloga velik vpliv (mediana 6) pri sprejemanju Scruma kot nove metodologije, kar se nanaša na raziskovalni cilj RC2.

Scrum precej olajša izvedbo zahtevnejših nalog, ki se po njegovih konceptih razdelijo na manjše uporabniške zgodbe in so zato bolj obvladljive. Pri implementiranju uporabniških zgodb si člani razvojne skupine pomagajo, saj so skupaj odgovorni za izdelano rešitev, kar dodatno olajša izvedbo nalog.

Način dela po Scrumu je skladen s potrebami posameznikov, saj menimo, da posamezniki cenijo to, da si lahko sami izbirajo naloge in se odločajo, kako jih bodo rešili, pri tem pa kontinuirano izdelujejo delujočo kodo, kar da posamezniku motivacijo za nadaljnje delo. To se je pokazalo tudi na obravnavanem projektu, ko so razvijalci začeli sami predstavljati izdelke na pregledih sprinta.

Scrum prinaša večje zadovoljstvo naročnika, saj lahko sproti testira izdelano rešitev in razvojni skupini podaja povratno informacijo. To mu omogoča, da dobi rešitev, ki jo potrebuje in ne nujno te, ki jo je definiral na začetku projekta. Slednje se je pokazalo tudi na obravnavanem projektu, če izpostavimo le en primer: razvojna skupina je sredi projekta na željo naročnika zamenjala v začetku razvito lastno rešitev za avtentikacijo uporabnika z rešitvijo si-pass.

5.3.3 VPLIV FAKTORJEV IZ SKUPINE POSAMEZNIK

Rezultate, ki se nanašajo na stopnjo difuzije faktorjev posameznika, prikazuje Tabela 9.

Tabela 9: Vpliv skupine faktorjev posameznik na stopnjo difuzije

Faktor	Povprečje	Mediana	Standardni odklon
Samostojno testiranje	3,87	4	1,51
Mreža osebnih stikov	3,87	4	1,41
Lastna pravila in nadzor nad lastnim delom	5,20	5	0,94
Učenje z delom	5,53	6	1,13

Rezultati kažejo, da se anketirancem noben izmed faktorjev skupine posameznik ne zdi nepomemben, saj za nobenega mediana ni 3 ali manj. Vprašani štejejo kot »niti pomembna niti nepomembna« dva faktorja (mediana 4), enega kot »nekoliko pomembnega« (mediana 5) in enega kot »pomembnega« (mediana 6).

Glede na raziskovalni cilj RC2, ti rezultati kažejo, da je faktor iz skupine posameznik, ki ima največji vpliv (mediana 6) na pripravljenost anketirancev za sprejem Scruma kot nove metodologije, učenje z delom.

Posameznik uporablja metodologijo Scrum, ker se jo lahko nauči in ovrednoti med praktičnim delom ter jo preprosto prilagodi svojemu načinu dela. Glede na to, da se je večina vprašanih prvič srečala z uporabo Scruma prav na tem projektu in pred tem ni imela praktičnih izkušenj, je faktor učenje z delom zagotovo pripomogel k večji sprejemljivosti Scruma. Scrum je okvir za razvoj programske opreme, zato ga posameznik lahko preprosto prilagodi svojemu načinu dela.

Anketiranci se niti ne strinjajo niti ne zanikajo, da jim prijatelji in kolegi večinoma zelo priporočajo uporabo metodologije Scrum. Razlog za to lahko najdemo v tem, da so anketiranci v svoji osebni mreži stikov med prvimi, ki uporabljajo to metodologijo.

5.3.4 VPLIV FAKTORJEV IZ SKUPINE OKOLJE

Tabela 10 prikazuje rezultate vprašalnika za stopnjo difuzije faktorjev okolja.

Tabela 10: Vpliv skupine faktorjev okolje na stopnjo difuzije

Faktor	Povprečje	Mediana	Standardni odklon
Kulturne vrednote	5,73	6	1,28
Tehnološka infrastruktura	5,93	6	1,28
Pravila skupnosti	4,87	5	0,74
Viri	4,47	5	1,30

Rezultati kažejo, da so vsi faktorji iz skupine okolje za anketirance »nekoliko pomembni« (mediana 5) ali »pomembni« (mediana 6), saj sta ti dve stopnji difuzije dosegla po dva faktorja.

Faktorja iz skupine okolje, ki imata največji vpliv (mediana 6) na pripravljenost anketirancev za sprejem Scruma kot nove metodologije sta: kulturne vrednote in tehnološka infrastruktura, kar se nanaša na raziskovalni cilj RC2.

Če okolje nudi vso potrebno tehnološko infrastrukturo in je Scrum skladen z vrednotami okolja, potem jo bo posameznik uporabljal. Ustrezna tehnološka oprema vsekakor pripomore k boljši uporabi Scruma, saj olajša marsikatero rutinsko delo, na primer SVN za vodenje različic programske opreme ali pa Jenkins za avtomatizacijo izgradnje verzij, kar je razvojna skupina obravnavanega projekta imela na voljo. Ker je v Scrumu veliko krajših sprintov, katerih cilj je delujoča koda, je taka tehnološka infrastruktura zelo pomembna, saj če ima skupina na voljo tako opremo, potem se lahko posveti razvoju rešitve in ne izgublja časa z ročno izvedenimi nalogami za podporo administrativnega dela. S pomočjo tehnologije lahko spremljamo tudi potek dela in upravljamo seznam zahtev, za kar je skupina Scrum uporabljala Jiro.

K večji uporabi Scruma pa pripomorejo tudi sledenje literaturi Scrum, izobraževanje in dostop do trenerja Scrum, saj s širšim znanjem Scruma posameznik najde smisel praks, ki jih Scrum zagovarja in se lahko poistoveti z njegovimi vrednotami. Posameznik s širšim znanjem Scruma tudi pravilno izvaja Scrum oziroma ga zna prilagoditi svojemu načinu dela tako, da še vedno sledi njegovemu okviru in ohranja koristi. Skupina Scrum je imela na voljo literaturo in certificiranje za skrbnika metodologije. Naj poudarimo, da sta bila zadnja dva skrbnika metodologije certificirana prav v času projekta. Podjetje je zaradi težav z uvajanjem Scruma

poiskalo pomoč pri izkušenem zunanjem izvajalcu, ki je pozneje postal tudi produktni vodja, to je PV2.

5.3.5 VPLIV FAKTORJEV IZ SKUPINE ORGANIZACIJA

Stopnjo difuzije faktorjev organizacije prikazuje Tabela 11.

Tabela 11: Vpliv skupine faktorjev organizacija na stopnjo difuzije

Faktor	Povprečje	Mediana	Standardni odklon
Medosebne mreže	4,80	5	1,26
Vrstniške mreže	5,20	5	1,15
Neformalna komunikacija	5,67	6	1,23
Tehnološke izkušnje	4,33	4	1,45
Delovne skupine	5,87	6	0,83
Mnenjski voditelji in zastopniki sprememb	5,07	6	1,62
Soodvisnosti od drugih	4,80	5	0,86
Tip posvojitelja	3,80	4	1,52
Hierarhija upravljanja	2,73	2	1,62

Rezultati vprašalnika o vplivih faktorjev skupine organizacija kažejo, da se zdi anketirancem en faktor »nepomemben« (mediana 2), dva faktorja »niti pomembna niti nepomembna« (mediana 4), trije faktorji »nekoliko pomembni« (mediana 5) in trije »pomembni« (mediana 6).

Glede na raziskovalni cilj RC2 ti rezultati kažejo, da so faktorji iz skupine organizacija, ki imajo največji vpliv (mediana 6) na pripravljenost anketirancev za sprejem Scruma kot nove metodologije, naslednji: neformalna komunikacija, delovne skupine ter mnenjski voditelji in zastopniki sprememb.

Zunanji vplivi za uporabo metodologije Scrum so majhni, prevladujejo notranje želje posameznikov, kar je spodbudno, saj nakazuje na to, da so posamezniki v Scrumu videli prednosti v primerjavi s prejšnjim načinom dela. Prednosti Scruma, ki jih razvijalci projekta cenijo, so: hitrost izdelave in kakovostnejša rešitev, zadovoljna stranka, povezanost skupine, pomoč sodelavcev in še bi lahko naštevali. Blizu pa so jim tudi koncepti Scruma: transparentnost, samoorganiziranost, samoiniciativnost in svoboda pri izbiri nalog. Prav svobodna izbira nalog prinaša zavezo, da jih bo posameznik dobro izdelal in posledično prevzel odgovornost zanje.

Posamezniki cenijo, da lahko sami načrtujejo delo in razrešujejo večino nastalih problemov in težav. Pri razreševanju problemov je zelo pomembna neformalna komunikacija, ki jo Scrum spodbuja, saj klasične specifikacije zahtev ni. V duhu agilnega manifesta [26] so v ospredju delujoča koda in uporabniške zgodbe namesto obsežne dokumentacije, ki so osnova za neformalni pogovor razvojne skupine. Glede na potrebe uporabniške zgodbe skupina razvijalcev sama načrtuje delo in skupaj razrešujejo probleme, s čimer se tudi breme problemov in odgovornosti porazdeli med člani skupine.

Glede na to, da je to v podjetju prvi projekt izveden po metodologiji Scrum, ga tudi sodelavci ne priporočajo v večji meri.

5.4 REZULTATI VPRAŠANJ ZA PRIMERJAVO S HIPOTEZAMI OVERHAGE-SCHLAUDERER

Tabela 12 prikazuje rezultate vprašalnika za preverjanje skladnosti z ugotovitvami Overhageja in Schlaudererja [14] glede dolgoročne sprejemljivosti Scruma in se nanašajo na raziskovalni cilj RC3. Osnova njunih hipotez je Rogersova teorija DOI.

Tabela 12: Vpliv skupine faktorjev inovacija na stopnjo difuzije (Overhage-Schlauderer)

Faktor		Povprečje	Mediana	Standardni odklon
Relativna prednost	čas do trga	5,93	6	1,03
	zahteve naročnika	5,40	6	1,06
	učinki učenja	5,80	6	1,01
	zadovoljstvo	5,73	6	1,16
Kompatibilnost	transparentnost	6,33	7	0,90
	sodelovanje	6,47	6	0,52
Kompleksnost	kompleksnost procesa	5,60	6	0,91
	disciplina	5,73	6	1,03

Rezultati vprašalnika o vplivih faktorjev skupine inovacija kažejo, da je sedem faktorjev za vprašane »pomembnih« (mediana 6) in en »zelo pomemben« (mediana 7), to je transparentnost.

Glede na rezultate lahko sklepamo, da uporaba metodologije Scrum omogoča boljše sodelovanje razvijalcev in večjo transparentnost procesa razvoja programske rešitve v primerjavi s tradicionalnim načinom, kar sta zaznala tudi Overhage in Schlauderer.

Boljše sodelovanje razvijalcev je pri Scrumu nujno, saj ni klasične podrobne specifikacije zahtev, ampak se mora razvojna skupina o rešitvi pogovoriti na podlagi uporabniške zgodbe, ki je napisana zelo na kratko in predstavlja izhodišče za pogovor. Poleg tega so razvijalci za izdelek skupno odgovorni in samoorganizirani, kar tudi zahteva boljše sodelovanje. Ko so člani razvojne skupine na obravnavanem projektu razumeli pomen praks Scrum in začeli uporabniške zgodbe obravnavati kot celote, se je tudi medsebojno sodelovanje močno izboljšalo, saj so se zavedali, da so skupaj odgovorni za izdelke.

Večja transparentnost procesa se kaže predvsem na dnevnih sestankih, kjer mora vsak razvijalec povedati, kaj je delal prejšnji dan in kaj bo delal danes, kar omogoča stalni pregled nad trenutnim stanjem razvoja rešitve. Transparentnost se dodatno kaže tudi na tabli Kanban, kjer se hitro opazijo ozka grla, kar so zaznali tudi člani skupine Scrum na obravnavanem projektu.

Podatki kažejo, da metodologija Scrum omogoča hitrejši razvoj, posameznik se nauči več in je bolj zadovoljen z razvito rešitvijo. Do enakih ugotovitev sta prišla tudi Overhage in Schlauderer.

Razvoj je pri Scrumu hitrejši, saj se večje zahteve razdelijo na manjše uporabniške zgodbe, ki so bolj obvladljive in posledično je manj napak oziroma se te hitreje odpravijo, kar je na dolgi rok hitreje in ceneje. Razvoj poteka v iteracijah in ob koncu vsake iteracije mora biti izdelek delujoč, kar pomeni, da je potrebno težave sproti razreševati, da dosežemo delujočo rešitev. Ko so razvijalci na obravnavanem projektu začeli obravnavati uporabniške zgodbe kot celote in jih sami predstavljali na pregledih sprintov, se je razvoj pohitril.

Posameznik si pri Scrumu razširi svoje znanje, saj mora za izdelavo zahtev sodelovati z drugimi razvijalci, da implementirajo celotno uporabniško zgodbo. Zaradi neformalne komunikacije in dnevnih sestankov posameznik velikokrat sliši za težave drugih, pri tem pa tudi rešitve zanje, kar še dodatno prispeva k širitvi znanja.

Povratna informacija naročnika je pri Scrumu zelo zaželena, tako razvojna skupina sproti dobiva informacijo o svojem delu in na podlagi tega lahko še izboljša svoje delo oziroma prilagodi rešitev naročnikovim potrebam, kar pa prinaša večje zadovoljstvo razvijalcev. Skupina Scrum je povratno informacijo naročnika na obravnavanem projektu začela dobivati zelo pozno, a ko jo je, ji je bila v veliko pomoč pri zaključevanju projekta. Razvijalec na pregledih sprinta vidi oziroma pokaže, kaj je v zadnjem sprintu izdelal, kar mu tudi prinaša stalno zadovoljstvo in pozitivno vpliva na večje sprejemanje Scruma.

5.5 REZULTATI SPECIFIČNIH VPRAŠANJ NA PODLAGI ZNAČILNOSTI PROJEKTA

Tabela 13 prikazuje rezultate, ki se nanašajo na trditve o delu na projektu prenove portala in zalednih sistemov in se nanašajo na raziskovalni cilj RC4.

Tabela 13: Vpliv karakteristik projekta na sprejetje Scruma

Faktor	Povprečje	Mediana	Standardni odklon
Lociranost skupine	6,67	7	0,62
Komunikacija	6,80	7	0,41
Predstavitev razvijalcev	5,67	6	1,54
Hitrost razvijanja	6,00	6	1,00
Zadovoljstvo stranke	6,40	6	0,63
Usposobljenost skrbnika metodologije	6,60	7	0,51
Usposobljenost produktnega vodja	6,27	7	0,96
Igra škatla KUDO	5,60	6	0,99
Igra *Best Sprint Task Force*	5,00	5	1,41
Tabla Kanban	5,40	6	2,13
Ocenjevanje uporabniških zgodb	5,20	6	1,47
Menjava metodologije	5,00	6	1,65
Sledenje metodologiji	5,07	6	1,67
Osebna rast	6,00	6	1,00
Prihodnost	6,53	7	0,64

Rezultati kažejo, da se vsi faktorji iz skupine karakteristik projekta zdijo pomembni, saj za nobenega mediana ni 4 ali manj. Vprašani štejejo enega kot »nekoliko pomembnega« (mediana 5), devet faktorjev kot »pomembnih« (mediana 6) in pet kot »zelo pomembnih« (mediana 7).

To, da skupina Scrum dela v isti pisarni, omogoča veliko spontane in neformalne komunikacije, kar prispeva k boljši kakovosti kode. Člani skupine Scrum tesneje sodelujejo kot pri tradicionalnih razvojnih projektih, saj morajo skupaj izdelati delujoče funkcionalnosti ob koncu sprinta, zato je možnost spontane in neformalne komunikacije nujna.

Velik pomen pri uspešni uporabi Scruma kot metodologije razvoja programske opreme ima tudi usposobljenost in osebna angažiranost skrbnika metodologije. To se je še posebej pokazalo na obravnavanem projektu, kjer se je šele po prihodu skrbnika metodologije SM3 začel vzpostavljati ritem razvoja, kot ga predpisuje Scrum.

Preskok iz slapovne metodologije na Scrum je po mnenju vprašanih dolgotrajen in terja precej napora, kar potrjuje dejstvo, da je Scrum preprost za razumeti in težak za implementacijo, saj zahteva ogromno discipline posameznikov [3, 23]. Težava pri uvedbi Scruma se je pokazala tudi na obravnavanem projektu, ko prakse Scrum sprva niso zaživele, kot jih predpisuje literatura.

Pomanjkanje stalne prisotnosti stranke, pri razvoju programske rešitve, močno vpliva na hitrost razvoja in končno zadovoljstvo z izdelkom, kar se je pokazalo tudi na tem projektu, saj je bila stranka vključena v proces Scruma precej pozno, to je ob prihodu produktnega vodja PV2. Ko se je stranka zadnje mesece pred produkcijo vključila na preglede sprinotv in začela testirati izdelke ter podajati sprotno povratno informacijo, se je tudi hitrost skupine povečala.

Agilne igre, kot sta na primer škatla KUDO [28] in *Best Sprint Task Force*, pomembno vplivajo na motivacijo, krepitev skupinskega duha in večjo samoorganiziranost skupine. Vsak posameznik si želi pohvale, saj mu ta sporoča, da delo dobro opravlja in da je nekdo opazil njegov trud ali pozitivno delovanje, kar je glavni namen škatle KUDO. To ga motivira in spodbuja, da bo svoje delo v prihodnje opravljal enako dobro ali še bolje. Na projektu je bilo izpolnjenih veliko kartic KUDO, kar je pripeljalo do večje pozornosti in samoiniciativne pomoči med člani skupine Scrum.

Uporaba metodologije Scrum je pozitivno vplivala na osebno rast in motivacijo posameznikov, zato je tudi želja, da bi v prihodnje razvijali po metodologiji Scrum, velika. Slednje je za dolgoročno sprejetost Scruma v podjetju zelo pomembno, saj so zainteresirani posamezniki pomembni pri uspešni vpeljavi novosti.

6 DELNO STRUKTURIRANI INTERVJUJI

Delno strukturirani intervjuji omogočajo poglobljeno razumevanje konteksta in zagotavljajo globlji vpogled v primerjavi z drugimi pristopi zbiranja podatkov [5], po drugi strani pa sledijo osnovni strukturi, kar omogoča primerljivost rezultatov [7].

V našo študijo smo jih na podlagi rezultatov vprašalnikov vključili z namenom, da bi preverili rezultate vprašalnika in tako dobili bolj zanesljive odgovore na zastavljena raziskovalna vprašanja. Pri oblikovanju strukture intervjuja smo izhajali iz povprečij posameznih odgovorov. Definirali smo izhodiščna vprašanja, ki so bila iztočnica za pogovor. Glede na odgovore pa smo vsakemu izmed intervjuvancev postavili še dodatna podvprašanja. Prav ta izhodiščna vprašanja nam omogočajo primerljivost odgovorov intervjuvancev.

Delno strukturirani intervjuji so bili izvedeni na manjši množici oseb kot vprašalniki, toda s skrbno izbranimi tipičnimi predstavniki članov skupine Scrum na projektu.

Rezultate smo za potrebe te študije tako zbrali na dva načina. Najprej smo s pomočjo vprašalnika zbrali preliminarne rezultate in jih obdelali. Vprašalnik je zajel širši krog oseb, ki so sodelovali na projektu. Posamezniki lahko interpretirajo vprašanja na različne načine, zato smo dobljene rezultate vprašalnikov dodatno preverili s pomočjo delno strukturiranih intervjujev. Lahko rečemo, da smo na dva načina potrdili rezultate in tako je študija bolj zanesljiva.

Vprašanja intervjuja so bila razdeljena na pet sklopov. Prvi sklop raziskuje razloge za tri najmanj sprejete prakse Scrum in se nanaša na RC1. Drugi sklop vprašanj išče potrditve, ali je pet faktorjev, ki najbolj vplivajo na sprejemljivost Scruma po oceni anketirancev, res pravi izbor in zakaj, to se nanaša na RC2. V tretjem sklopu iščemo (ne)strinjanje s hipotezami Overhage-Schlauderer, da lahko dosežemo RC3. V četrtem sklopu se osredotočimo na specifike dela na projektu in v petem sklopu na negativne in pozitivne vidike projekta s stališča Scrum metodologije. Zadnja dva sklopa nam pomagata odgovoriti na RC4.

6.1 STOPNJA SPREJETJA KLJUČNIH PRAKS SCRUM

Vse prakse so za Scrum enako pomembne, če ga želimo dobro izvajati, zato smo se pri intervjujih osredotočili na tri prakse, ki so bile s pomočjo vprašalnikov zaznane, da so od naštetih na projektu najmanj sprejete:

- skupinsko ocenjevanje uporabniških zgodb po metodi *Team Estimation Game*,
- načrtovanje izdaje na osnovi ocenjene hitrosti razvojne skupine,
- uporaba diagrama Burndown za nadzor poteka dela.

Za vsako izmed treh najmanj sprejetih praks smo intervjuvanca vprašali:

1. Ali se strinjaš, da je bila [praksa] ena izmed najmanj sprejetih praks na projektu?
    a. Če se, ali se ti zdi pomembno, da bi izboljšali njeno sprejemljivost? Kako?
    b. Če se ne strinjaš, zakaj?
2. Bi opozoril na katero drugo prakso, ki je bila po tvojem mnenju slabše sprejeta kot te tri in je pomembna za uspešno delo na projektu?

6.2 FAKTORJI, KI VPLIVAJO NA SPREJEMLJIVOST SCRUM

Na dolgoročno sprejemljivost Scruma vpliva vrsta faktorjev. Vprašalnik je pokazal, da ima na projektu največji vpliv za uporabo Scruma naslednjih 5 faktorjev:

- **kompatibilnost**
  Trditev: metodologijo Scrum uporabljam, ker se veliko bolj ujema z mojim želenim načinom dela, vrednotami in izkušnjami kot metodologije, ki sem jih uporabljal prej.
- **relativna prednost**
  Trditev: metodologijo Scrum za razvoj programske opreme uporabljam, ker je boljša od metodologij razvoja programske opreme, ki sem jih uporabljal prej.
- **odpor uporabnika**
  Trditev: metodologijo Scrum uporabljam, ker pri razvoju programskih rešitev olajša izvedbo zahtevnejših nalog.
- **tehnološka prednost**
  Trditev: metodologijo Scrum uporabljam, ker je veliko bolj napredna od načinov razvoja programske opreme, ki sem jih uporabljal prej.
- **tehnološka infrastruktura**
  Trditev: za uporabo metodologije Scrum imamo na voljo vso potrebno tehnološko infrastrukturo.

Za vsakega izmed petih faktorjev, ki imajo na projektu največji vpliv, smo intervjuvanca vprašali:

1. Ali se strinjaš, da je imel [faktor] na projektu največji vpliv za uporabo Scruma?
    a. Zakaj se strinjaš s to trditvijo? Če se (ne) strinjaš, zakaj?
2. Ali bi izpostavil še katere druge faktorje, ki so po tvojem mnenju najbolj vplivali na sprejemljivost Scruma na projektu? Zakaj oz. navedi izkušnjo/primer?

6.3 HIPOTEZE OVERHAGE-SCHLAUDERER

Za potrebe primerjave hipotez Overhage-Schlauderer smo preverili strinjanje z vsemi osmimi hipotezami, ki sta jih postavila:

- Metodologija Scrum omogoča hitrejši razvoj programske opreme kot klasične metodologije.
- Metodologija Scrum omogoča razvoj programske opreme, ki bolj ustreza zahtevam naročnika kot klasične metodologije.
- Metodologija Scrum omogoča, da se pri razvoju programske rešitve naučimo več kot pri klasičnih metodologijah.
- Metodologija Scrum omogoča večje zadovoljstvo razvijalcev z razvito programsko rešitvijo kot klasične metodologije.
- Metodologija Scrum omogoča boljši pregled nad potekom razvoja programske rešitve kot klasične metodologije.
- Metodologija Scrum omogoča boljše sodelovanje razvijalcev pri razvoju programske rešitve kot klasične metodologije.
- Metodologija Scrum zmanjša kompleksnost procesa razvoja programske rešitve kot klasične metodologije.
- Metodologija Scrum zahteva več discipline pri razvoju programske rešitve kot klasične metodologije.

Za vsako hipotezo smo intervjuvanca vprašali:

1. Ali se strinjaš s [hipotezo]? Zakaj se (ne) strinjaš s [hipotezo]?
2. Kako se je to pokazalo na vašem projektu?

6.4 ZNAČILNOSTI PROJEKTA

V zvezi z značilnostmi projekta smo izmed rezultatov vprašalnika izbrali tiste, ki bi jih po našem mnenju veljalo obdržati oziroma izboljšati pri naslednjem projektu.

Intervjuvance smo vprašali:

1. Ste imeli s konceptom »done« uporabniških zgodb težave? Če da, kako ste jih rešili?
2. Zakaj se ti zdi vloga skrbnika metodologije (ang. *ScrumMaster*) tako pomembna?
3. Je naročnik po tvojem mnenju zadovoljen z rešitvijo? Je polno sodeloval pri projektu? Če ni, bi si želel, da bi bil bolj prisoten pri projektu?
4. Zakaj se ti zdi fizična tabla Kanban pomembna poleg elektronske v Jiri?
5. Se strinjaš, da je ocenjevanje v točkah (ang. *story points*) bolj objektivno kot v urah? Zakaj?
6. Ali je na prvem projektu razvoja programske opreme po metodologiji Scrum bolje uvesti vse prakse Scrum naenkrat ali postopoma?

6.5 POSTMORTEM ANALIZA

V skladu s priporočili postmortem analize [1] smo raziskali še pozitivne in negativne vidike projekta s stališča Scruma.

Intervjuvanca smo vprašali:

1. Kaj bi s stališča metodologije Scrum izpostavil kot najbolj pozitivno plat projekta?
2. Kaj je bilo po tvojem mnenju s stališča metodologije Scrum slabo in bi bilo treba pri naslednjem projektu izboljšati? Kako bi vpeljal potrebne izboljšave?

7 REZULTATI IN ANALIZA INTERVJUJEV

Intervjuje smo v sredini maja 2016 opravili s predstavniki skupine Scrum, ki so izpolnili vprašalnike o sprejetosti Scruma na obravnavanem projektu. Intervjuvali smo pet najbolj kompetentnih predstavnikov in vsi so zaposleni v podjetju, ki je izdelalo programsko rešitev. Izmed vseh intervjuvancev je ena oseba na projektu nastopala v vlogi skrbnika metodologije, v nadaljevanju SM, ostali štirje v vlogi razvijalcev, v nadaljevanju R1–R4.

Dva izmed intervjuvancev sta prvič sodelovala pri razvoju programske opreme (ang. *Junior Developer*) in zato nimata izkušenj z drugimi metodologijami razvoja, druga dva intervjuvanca sta že delala po Scrumu v podjetjih, kjer sta bila prej zaposlena, en intervjuvanec je pred tem projektom delal po slapovni metodologiji in je bil na tem projektu prvič seznanjen z metodologijo Scrum.

Posamične intervjuje smo izvedli v prostorih podjetja in so trajali od 50 do 90 min. Med samim intervjujem smo si zapisovali odgovore in jih takoj po intervjuju uredili. V skladu s priporočili dobre postmortem analize [1] smo urejeni zapis intervjuja dali v pregled intervjuvancu, da preveri, ali smo pravilno razumeli njegova mnenja oziroma odgovore.

7.1 REZULTATI STOPENJ SPREJETJA POSAMEZNIH PRAKS SCRUM

Rezultati prvega dela intervjuja se nanašajo na RC1.

Vprašalnik je pokazal, da so metoda *Team Estimation Game*, načrtovanje izdaje in diagram Burndown najmanj sprejete prakse na projektu. Da bi ugotovili, ali je to res in našli morebitne vzroke za to, smo o tem povprašali intervjuvance.

To, da je praksa ***Team Estimation Game*** za ocenjevanje uporabniških zgodb najmanj sprejeta, je razumljivo, saj se skupina na projektu z njo ni srečala: »*Nismo je izvajali na projektu*« (SM, R1, R3). »*Za ocenjevanje smo uporabljali Planning Poker*« (R4).

Mnenja o smotrnosti izboljšanja sprejemljivosti prakse *Team Estimation Game* so deljena, saj sta bili na projektu uporabljeni ocenjevanji uporabniških zgodb v urah in točkah po metodi *Planning Poker*. Dva izmed vprašanih bi si želela preizkusiti to prakso, ker »*pri Planning*

Pokerju ocenjuješ zgodbo za zgodbo in ko jo oceniš, se ne vračaš k popravkom ocene« (R4). Drugi je mnenja, da »*je uporabna, ko imaš večje zgodbe, to je na začetku projekta, da dobiš bolšji pregled nad celotno vsebino*« (SM). Dve osebi ne vidita potrebe, da bi njeno sprejemljivost izboljšali: »*Najbolj blizu mi je ocenjevanje po dejanskem času porabe*« (R2, R3). »*Težavnost naloge je lahko minimalna, a lahko zanjo porabiš veliko časa, zato se mi ocenjevanje v točkah ne zdi smiselno, saj na drug način predstaviš čas porabe za izdelovanje posamezne naloge*« (R3).

Glede **načrtovanja izdaje na osnovi ocenjene hitrosti skupine** sta se pokazali dve stvari, ki prispevata k nižji sprejemljivosti te prakse. Prva stvar je ta, da skupina izdaj ni načrtovala: »*Izdaj nismo načrtovali […]nekajkrat smo izdelke več sprintov zapakirali v izdajo*« (R1). »*Izdaj nismo planirali, samo sprinte*« (R4) oziroma, da je načrtovala le dve izdaji »*Imeli smo samo dve izdaji, sicer smo izvajali sprint planning*« (R2). »*Jeseni 2014 […] in januarja 2015 smo imeli planiranje izdaje, vmes pa je PV1 na ločenih sestankih planiral s posamezniki skupine*« (SM). Odgovori kažejo na to, da skupina na začetku ni čisto ločila planiranja izdaj od planiranja sprintov ali pa so na to prakso preprosto pozabili, saj se je izvajala le na začetku projekta. Druga stvar je osnova za načrtovanje izdaje, ki sprva ni bila hitrost skupine: »*Izdaja je vsebovala zgodbe, ki jih je naročnik želel, ne glede na hitrost skupine, ampak bolj po občutku PV1*« (R2). »*Ker je bilo vedno preveč naplaniranega*, *kot je bila skupina sposobna izdelati, smo januarja 2015, na željo PV1 in stranke, začeli meriti velocity in sprint planning je postal tudi relase planning*« (SM).

Rezultati vprašalnika so pokazali, da je sprejetost prakse **diagram Burndown** za nadzor poteka dela zelo nizka, a je za sledenje napredku projekta pomembno, da skupina ve, kje se nahaja, zato smo želeli raziskati, zakaj se ni uporabljala. Intervjuji so pokazali, da je bila opuščena iz dveh razlogov. Prvi razlog je nedosledno vpisovanje ur: »*Na začetku smo jo izvajali, a se ni obnesla, saj nismo bili dosledni pri vpisovanju ur*« (R2), drugi razlog je demotivacija skupine ob prepočasnem spuščanju grafa: »*Diagram ni dajal motivacije, saj se je prepočasi spuščal. Ocena naloge je bila velikokrat nižja kot dejanska poraba časa za posamezno nalogo in ob koncu sprinta smo prišli do sredine planiranega časa in ne do 0. Po šestih sprintih smo to opustili, saj nismo videli pozitivnega učinka*« (R4).

Mnenja glede njene ponovne uporabe so deljena, eni ne vidijo potrebe po njej: »*Ne zdi sem mi potrebna, saj je informacij o poteku dela dovolj na dnevnem sestanku, tabli Kanban in v Jiri*« (R1), drugi pa »*Zdi se mi pomembna, ker vidiš, kje so težave, jih odpraviš in tako pohitriš razvoj. Je transparentna*« (R3). »*To prakso bi si želel ponovno uvesti in sem pripravljen sodelovati vsak dan z vpisom ur, želim pa si, da bi ScrumMaster na začetku skrbel za red in disciplino glede tega, da nam preide v navado*« (R2). »*SM3 je diagram sam spremljal preko Jire in po potrebi skupino spodbudil, če je bila prepočasna*« (SM).

Poleg prej omenjenih treh najmanj sprejetih praks so intervjuvanci omenili še težave s **konceptom »done«**: »*Nekaj časa smo se držali pravil koncepta »done«, ker nas je SM3 opozarjal na to, potem pa ne več. Nismo bili preveč dosledni in samoorganizirani. Morali bi*

dalj časa vztrajati, da bi nam prešlo v navado« (R2). »*Biti bi morali bolj dosledni pri njegovi uporabi. Koncept je pomemben, da veš, ali je naloga res zaključena in delujoča, saj ima velik vpliv na kakovost kode*« (R1) in začetne težave s fokusom na **dnevnih sestankih**, ki »*na začetku niso bili v duhu Scruma, saj niso imeli fokusa na to, kaj si delal prejšnji dan, kaj boš danes in kje imaš probleme, bilo je prosto po Prešernu*« (R2). Izpostavljena je bila tudi praksa **grupiranje uporabniških zgodb po funkcionalnostih in sprintih**: »*Prišla je do izraza proti koncu projekta. Prej pa je bila narava razvoja taka, da se v določenih fazah pokrije vse uporabniške zgodbe izbranega epica, ne pa najpomembnejše uporabniške zgodbe različnih epicov*« (R4).

7.2 REZULTATI FAKTORJEV, KI VPLIVAJO NA SPREJEMLJIVOST SCRUMA

Na dolgoročno sprejemljivost Scruma vpliva vrsta faktorjev, kar je povezano z RC2. Vprašalnik je pokazal, da ima za uporabo Scruma na projektu največji vpliv naslednjih pet faktorjev: ujemanje s posameznikovim načinom dela, vrednotami in izkušnjami (kompatibilnost); je boljši kot druge metodologije (relativna prednost); olajša izvedbo zahtevnejših nalog (odpor uporabnika); je veliko bolj napreden (tehnološka prednost) in razpoložljivost potrebne tehnološke infrastrukture (tehnološka infrastruktura). Povprečne vrednosti odgovorov vprašalnika za te faktorje so bili od 6,33 do 5,93 od možnih 7.

Intervjuvanci so za faktor **kompatibilnosti**, to je ujemanje z njihovim želenim načinom dela, vrednotami in izkušnjami potrdili, da se strinjajo, da Scrum to omogoča, zato ga tudi uporabljajo. Glede načina dela so izpostavili: »*Razbijanje večjih zahtev na manjše in reševanje posameznih manjših delov […] skupinsko delo te razbremeni odgovornosti, ker se porazdeli med vsemi člani in ko imaš težave, se lahko zaneseš na skupino*« (R2). »*Ni vse vnaprej določeno z dokumentacijo, omogoča samostojno delo, svobodno izbiro načina izdelave posamezne naloge […] več komuniciranja z ostalimi člani skupine*« (R3). »*Zahteve so vsebinsko in časovno bolj dorečene, funkcionalnosti so razbite na manjše naloge, razvoj poteka v več krajših ciklih*« (R1). »*Vsak sprint prinese nekaj delujočega […], kar ti daje večjo motivacijo, zagon in ob uspešno zaključenem delu tudi zadovoljstvo*« (R4). Izmed vrednot, ki jih Scrum zagovarja, so izpostavili tiste, ki se skladajo z njihovimi vrednotami, to so: »*Samostojnost, kreativnost, samoiniciativnost, odgovornost in transparentnost*« (SM). »*Samoorganiziranost, samoiniciativnost in pomoč med člani skupine*« (R4). »*Skupinsko delo in transparentnost procesa razvoja*« (R1). »*Večja pripadnost skupini in večja zavzetost pri izdelavi dobre rešitve*« (R3).

Za faktor **relativne prednosti** so bili intervjuvanci deljenih mnenj, da je Scrum boljši kot metodologije, ki so jih uporabljali prej, saj nekateri tradicionalnih metodologij za razvoj programske opreme še niso uporabljali: »*Težko ocenim, saj z drugimi metodologijami nimam izkušenj*« (R2). »*Po waterfallu nisem nikoli delal, zato težko komentiram*« (R1). Oziroma so

vedno delali na agilen način: »*Nikoli nisem delala po waterfallu, zato težko primerjam iz prakse, lahko samo kot opazovalka drugih projektov, ki delajo po waterfallu*« (SM). Pri tem je potrebno poudariti, da sta dva izmed intervjuvancev prvič sodelovala pri razvoju programske opreme (ang. *Junior Developer*) in zato nimata izkušenj z drugimi metodologijami razvoja, druga dva intervjuvanca pa sta že delala po Scrumu v podjetjih, kjer sta bila prej zaposlena. Tisti, ki že imajo izkušnje s tradicionalnimi metodologijami, pa so se strinjali, da je Scrum boljši, ker »*Omogoča večji uspehi projekta, saj je skupina bolj povezana, ni monotonosti, spremembe se lažje upravljajo*« (R3). »*Pravočasno lahko zaznaš morebitne nedoslednosti v prvotnem načrtu in se temu hitreje prilagodiš*« (R4).

Glede faktorja **odpor uporabnika** so se vsi strinjali, da Scrum olajša izvedbo zahtevnejših nalog, saj: »*So ti na voljo drugi člani ekipe in ti pomagajo, če se ti zatakne ter veš, da lahko računaš nanje*« (R3). »*V primeru težavnejših nalog dobiš pomoč drugih – na dnevnem sestanku, ko poveš, da imaš problem ali ko nekdo zazna, da se zadeve odvijajo v napačno smer in te opozori na to – to zadnje se mi je na projektu večkrat zgodilo in sem vesel, da nisem izgubljal časa s stvarmi, ki ne gredo v pravo smer*« (R4). »*Nalogo lahko razdeliš na več manjših nalog, ki so bolj obvladljive*« (R2) in manjše naloge »*se porazdelijo med več člani ekipe*« (R4). »*Scrum te sili, da funkcionalnosti razbiješ na manjše dele in lažje se fokusiraš na več manjših problemov, kot na enega velikega*« (SM).

O faktorju **tehnološke prednosti** se niso vsi strinjali s trditvijo, da je Scrum bolj napredna kot metodologije, ki so jo prej uporabljali, saj je za nekatere to prvi projekt razvoja programske opreme. Tisti, ki imajo več izkušenj z razvojem programske opreme, pa so se strinjali, da je Scrum naprednejši, saj: »*Se mi zdi bolj primerna za razvoj programske opreme, ker so spremembe edina stalnica in s Scrumom se hitreje in enostavneje nanje odzoveš*« (SM). »*Danes moraš biti učinkovit, hiter, dosleden in v kratkem času narediti čim več, če hočeš biti produktiven in s Scrumom to lahko dosežeš*« (R3). »*Scrum omogoča hitrejši reakcijski čas in spremljanje postopka razvoja, to je transparentnost*« (R4).

Glede faktorja **tehnološke infrastrukture** so se vsi razen enega strinjali, da imajo vse kar potrebujejo, da lahko razvijajo rešitev po metodologiji Scrum: »*Imamo SVN*[2]*, Confluence*[3]*, Jira, Jenkins, kar nam vsekakor olajša delo po Scrumu*« (R2). »*Jira, za podporo procesa metodologije. Jenkins – aplikacija za avtomatizacijo verzij, kar omogoča hitrejšo verzijo produkta vsak sprint*« (R1). »*Imamo skupno sobo, kjer lahko mirno komuniciramo med seboj in s tem ne motimo drugih skupin; prostor za veliko tablo Kanban in Jiro. Imamo tehnično okolje za continuous integration (SVN, Jenkins)*« (SM). En razvijalec pa trdi, da je: »*Scrum neodvisen od tehnološke infrastrukture*, *saj tehnološka infrastruktura nima veze z metodologijo razvoja*« (R3).

[2] Apache Subversion, pogosto imenovan SVN, je programska oprema za vodenje različic in kontrolni pregled sistema. Razvijalci programske opreme ga uporabljajo za ohranitev trenutne in zgodovinske različice datotek izvorne kode in dokumentacije.
[3] Orodje za organizacijo dela, kreiranje dokumentov in diskusijo med člani skupine.

Poleg teh faktorjev, za katere so vprašalniki pokazali, da imajo največji vpliv pri uporabi Scruma in s katerimi so se vsi intervjuvanci strinjali, so posamezniki izpostavili še nekaj drugih faktorjev, ki imajo po njihovem mnenju velik in pozitiven vpliv za uporabo Scruma. Dve osebi sta izpostavili faktor **neformalne komunikacije**, češ da Scrum omogoča veliko spontane in neformalne komunikacije med člani razvojne skupine, kar omogoča, da: »*hitreje prideš do rešitve, saj veš, kdo kaj zna in kdo lahko katero nalogo naredi oz. kdo ti lahko pri čem pomaga*« (R2) in »*da sem bolj povezan s člani skupine*« (R3). Dva sta izpostavila faktor **vrstniških mrež**, kar pomeni, da Scrum uporabljajo, ker programsko rešitev razvijajo v skupini s sodelavci, s katerimi se tudi sicer veliko družijo: »*vedno lahko računaš na pomoč skupine, če se ti zatakne. V naši skupini so bili vsi pripravljeni pomagati*« (R1). Oba ta dva faktorja sta tudi pri vprašalniku dobila visoke ocene, pri prvem je povprečje odgovorov 5,67, pri drugem 5,2. Dve osebi sta izpostavili tudi faktor **delovnih skupin**, kar omogoča, da sami načrtujemo delo in sami razrešimo večino problemov in težav: »*Scrum spodbuja samoiniciativnost in samoorganiziranost. Končni cilj je, da se oblikuje samoorganizirana skupina, kar je seveda odvisno od ljudi. V tej skupini so ljudje, ki jim takšen način dela ustreza in so sprejeli metodologijo Scrum*« (SM). »*Skupaj smo prišli do boljše rešitve, saj posameznik sam to težko doseže. Dobro je, da si sami izbiramo naloge in njihov način izdelave, saj si lažje optimiziraš delovni proces in tako dosežeš večji učinek pri kvaliteti kode in boljše odnose v skupini*« (R1). Tudi za ta faktor se je pri vprašalnikih izkazalo, da je bil zelo dobro sprejet, saj je povprečje odgovorov 5,87.

Izpostavljen je bil tudi faktor **potrebe posameznika po prepoznanju**, kar pomeni, da je način dela, ki ga Scrum predpisuje, skladen s posameznikovimi potrebami: »*Všeč mi je, da dobim problem in sem lahko pri njegovem reševanju kreativna*« (SM). Tudi ta faktor je med anketiranci dobil visoko povprečje, to je 5,73. Omenjen je bil tudi faktor **razreševanja problemov**, kjer Scrum kot metodologija razrešuje veliko problemov: »*pri waterfallu lahko veliko časa izgubiš na stvareh, ki niso pomembne oz. se kasneje izkažejo za napačne, torej pomanjkanje sprotne povratne informacije*« (R4) in ta sprotna povratna informacija omogoča sprotno razreševanje problemov. Tudi ta faktor so anketiranci ocenili precej visoko, saj je povprečje doseglo 5,53. Izpostavljen je bil tud faktor **samostojnega eksperimentiranja**: »*imaš možnost preizkušanja in lahko predlagaš kaj boljšega*« (R2), ki pa pri večini anketirancev ni bil toliko izpostavljen, saj rezultati vprašalnikov kažejo, da je povprečje le 3,87. Ena oseba je poudarila, da je zanjo pomemben tudi faktor **standarda**, saj Scrum uporablja, ker postaja standardna metodologija za razvoj programskih rešitev: »*Danes je agilen razvoj trend, saj izvajalci želijo optimizirati proces razvoja programske opreme, saj se strankam spreminja način dela, ker se jim zahteve hitreje spreminjajo*« (R4). Slednji faktor ni dobil posebne teže pri vprašanih anketirancih, saj je povprečje odgovorov anketirancev le 4,93.

Če povzamemo, ankete so pokazale prave rezultate, ki so jih intervjuji samo še podkrepili s konkretnimi pojasnili in primeri, zakaj je posamezni faktor imel velik vpliv pri uporabi

Scruma. Pri dolgoročni sprejemljivosti Scruma moramo biti pozorni na te faktorje, saj povečujejo njegovo dolgoročno sprejemljivost.

7.3 REZULTATI VPRAŠANJ ZA PRIMERJAVO S HIPOTEZAMI OVERHAGE-SCHLAUDERERJA

Da bi lahko primerjali rezultate, ki sta jih Overhage in Schlauderer dobila pri svoji raziskavi [14], kar je cilj RC3, smo intervjuvancem postavili vprašanja o hipotezah, ki sta jih postavila. Že sam vprašalnik je pokazal dokaj visoko strinjanje s temi hipotezami, saj so bili odgovori visoko ocenjeni, v povprečju od 6,47 do 5,4 od 7 možnih. Z intervjujem smo želeli preveriti razloge zanje.

Prva hipoteza: Metodologija Scrum omogoča **hitrejši** razvoj programske opreme kot klasične metodologije. Rezultati intervjuja potrjujejo, da razvijalci zaznavajo hitrejši razvoj programske opreme po metodologiji Scrum, kar predstavlja korist pri sprejemanju nove metodologije. Na eni strani intervjuvanci razloge za večjo hitrost vidijo v komunikaciji: »*Zaradi dnevnega poročanja na sestankih je komunikacija v skupini boljša in hitro se vidi napredek oziroma zastoj skupine, tako imamo možnost ukrepanja tekom samega razvoja*« (R2). »*Na standupih dnevno vidiš, kje so ozka grla in energijo usmeriš v to, da sproti odpravljaš probleme*« (R3). »*Dela razvijalcev se prepletajo, vsi vedo za dela drug drugega in se tudi sproti usklajujejo pri morebitnih povezavah, npr. baznik z backend razvijalcem. Na dolgi rok je manj napak, saj jih hitreje odkriješ in odpraviš.*« (R1). Na drugi pa v drobljenju večjih funkcionalnosti na manjše sklope: »*več manjših uporabniških zgodb lahko hitreje naredimo, testiramo in vidimo ali gremo v pravo smer. Če ugotovimo, da se nekaj ne bo izšlo, iščemo drugačne rešitve. Pri waterfallu se razvoj začne šele po ustrezno končani analizi in vseh zahtev ni mogoče dobro popisati, saj stranka na začetku velikokrat ne ve točno kaj si želi oz. se zahteve spremenijo*« (SM). »*Delaš več manjših stvari*« (R1). Odgovori naših intervjuvancev potrjujejo Overhagejevo in Schlaudererjevo hipotezo. Prišli smo do podobnih ugotovitev kot onadva, to so hitrosti reakcije na spremembe zahtev, ki se pojavijo med samim razvojem in spremembe prioritet zahtev. Po teoriji Scruma naročnik dobi tiste funkcionalnosti, ki so zanj uporabne in ne nujno tiste, ki so bile podane v začetku projekta.

Druga hipoteza: Metodologija Scrum omogoča razvoj programske opreme, ki **bolj ustreza zahtevam naročnika** kot klasične metodologije. Intervjuvanci se strinjajo s to hipotezo, zato se šteje kot prednost, ki spodbuja sprejetje nove metodologije. Pri tem je večina poudarila obvladovanje sprememb, ki se pojavijo v času razvoja in s tem bolj ustrezne rešitve za stranko: »*Naročnik sproti preverja izvedbo svojih zahtev in ima možnost dajati povratno informacijo*« (R1). »*Zahteve se lahko vmes spreminjajo in naročnik dobi tako rešitev, ki jo potrebuje in ne tisto, ki jo je definiral na začetku*« (R3). »*Ker dobiva izdelek v iteracijah […] ima naročnik možnost izboljšati produkt sebi v prid tekom razvoja*« (R4). »*Stranka na začetku*

ne more podati vseh zahtev in predvideti vseh sprememb, kar je pri waterfallu naknadno težko implementirati oz. je drago in za stranko neugodno, saj dobi stranka rešitev precej pozno. Pri Scrumu lahko podaja povratno informacijo sproti« (SM). Naši intervjuvanci so potrdili hipotezo Overhageja in Schlaudererja. Njuni intervjuvanci so našteli zelo podobne razloge za to, da Scrum omogoča večjo ustreznost zahtev stranke kot naši: učinkovito obvladovanje sprememb zahtev med razvojem in zgodnja povratna informacija stranke o smeri razvoja, ki prinaša uporabno in delujočo rešitev oziroma tako rešitev, ki jo stranka potrebuje.

Tretja hipoteza: Metodologija Scrum omogoča, da se pri razvoju programske rešitve **naučimo več** kot pri klasičnih metodologijah. Intervjuvanci so to hipotezo potrdili in zato lahko rečemo, da je boljši učni učinek prednost in motivator pri sprejemu Scruma: »*Scrum zahteva interdisciplinarnost članov skupine. Vsak član je sicer specialist v nečem, npr. za bazo, a skozi komunikacijo, ki jo Scrum zahteva, se specialistična znanja vse bolj prepletajo in pride do prenosa znanja med različnimi področij. Vsi si razširijo znanja*« (SM). »*Povezan si s celo skupino in nisi vpet samo na svoje področje. Ko slišiš za težave drugih, sliši tudi rešitve in si ob tem razširiš znanje*« (R4). »*Naučil sem se testiranja, saj je ob koncu sprinta potrebno verzijo testirati; bolje uporabljati SVN; Confluence, Jiro in več samodiscipline*« (R2). »*Vsak bi moral poznati vsa področja. Si sicer specialist za eno področje, a moraš poznati tudi druga področja, če želiš implementirati dobro rešitev, zato je po mojem mnenju rešitev boljša kot pri waterfallu in na koncu imaš manj težav z vzdrževanjem in razširljivostjo produkta*« (R3). Ti odgovori se ujemajo z rezultati Overhageja in Schlaudererja. Pri odgovorih pa njuni intervjuvanci omenjajo tudi prenos znanja naročnika, kar naši intervjuvanci ne omenjajo direktno, saj je bilo neposredno sodelovanje naročnika z razvojno skupino na projektu zelo okrnjeno. Vsebinsko znanje je bilo preneseno na člana razvojne skupine preko produktnega vodja: »*Pri Scrumu mora razvijalec dobro poznati vsebino in to po vseh nivojih*« (R1).

Četrta hipoteza: Metodologija Scrum omogoča **večje zadovoljstvo** razvijalcev z razvito programsko rešitvijo kot klasične metodologije. Rezultati intervjujev potrjujejo, da so razvijalci bolj zadovoljni z razvito rešitvijo po metodologiji Scrum. Povečano zadovoljstvo vidimo kot prednost, ki pozitivno vpliva na sprejem Scruma: »*Izdelek je boljši, ker se sproti usklajujemo o tem kaj in kako delati, sproti odpravljamo težave in posledično je tudi zadovoljstvo večje*« (R2). »*Pozitivna kritika sodelavcev pri posameznih nalogah, z namenom izboljšanja rešitve, tj. sproti med samim razvojem in na pregledih sprinta*« (R4). »*Na pregledih sprinta […] pokažeš svoje delo in to prinaša zadovoljstvo. Krajše iteracije omogočajo večkratni pregled izdelanega, kar prinaša konstantno zadovoljstvo*« (R1). »*Razvijalec vidi svoj doprinos k celotni uporabniški zgodbi in zaradi tega je bolj zadovoljen z rešitvijo, saj npr. ne naredi neke procedure ne da bi vedel, kje/zakaj se uporablja, kot je to pogosto pri waterfallu. Pri nas se je to pokazalo na pregledih sprinta, ko so razvijalci prav s ponosom pokazali, kaj so naredili v tekočem sprintu*« (SM). »*Ker je pripadnost skupini in projektu večja*« (R3). Naši intervjuvanci so potrdili hipotezo Overhageja in Schlaudererja. Njuni intervjuvanci so našteli zelo podobne razloge za to, da Scrum omogoča večje zadovoljstvo razvijalcev z razvito rešitvijo kot naši: želja po dobro izdelani rešitvi s pomočjo

Scruma je večkrat izpolnjena, saj ti skupina pomaga doseči ta cilj, posledično je ponos in zadovoljstvo večje, saj je tudi rešitev boljša, ker pa so iteracije kratke, je zadovoljstvo konstantno.

Peta hipoteza: Metodologija Scrum omogoča **boljši pregled nad potekom razvoja** programske rešitve kot klasične metodologije. Intervjuvanci so se strinjali s to hipotezo, zato lahko rečemo, da je transparentnost poteka razvoja pri Scrumu večja kot pri tradicionalnih metodologijah: »*Na dnevnem sestanku se kaže dnevni napredek, kjer en drugega lahko opomnimo, vprašamo za pomoč oz. pojasnimo problem, na pregledu sprinta se vidi, kaj se je naredilo v zadnjem sprintu*« (R1). »*Dnevni sestanek, kjer je vse transparentno, saj se vidi kdo ima preveč, kdo premalo dela in kdo ima težave. Dobiš občutek ali boš res dosegel tisto, kar si si zadal*« (SM). »*Dnevni sestanek, planiranje in pregled sprinta omogočajo boljši pregled. Imeli smo naloge, ki so se vlekle iz sprinta v sprint […] če posameznik ni prosil za pomoč, je skrbnik metodologije pozval ekipo, da naj mu pomaga*« (R4). Ti odgovori se ujemajo z rezultati Overhageja in Schlaudererja. Njuni intervjuvanci so prav tako omenjali transparentnost, ki je vidna na dnevnih sestankih in posledično omogoča hitro ukrepanje ob zaznanih težavah, kar je vsekakor prednost Scruma.

Šesta hipoteza: Metodologija Scrum omogoča **boljše sodelovanje razvijalcev** pri razvoju programske rešitve kot klasične metodologije. Rezultati intervjujev kažejo, da se sodelovanje razvijalcev pri razvoju programske rešitve po metodologiji Scrum poveča, kar je vsekakor prednost pred tradicionalnimi načini razvoja: »*Zaradi multidisciplinarnosti razvijalcev, lažje najdeš nekoga, ki ti svetuje, ko naletiš na problem, tj. skupinsko odpravljanje problemov […] dnevni sestanek te prisili, da komuniciraš*« (R1). »*Na pregledih sprintov se vidi, da smo skupaj pripeljali zgodbe do konca. Na retrospektivah smo slišali stvari, ki jih sicer ne bi, saj je imel vsak možnost povedati, kaj ga pri delu ovira in spodbuja, kar je za izboljšanje dela zelo pomembno. Igra jadrnica je vizualizacija retrospektive in zagotavlja sproščenost članov, da lažje izrazijo stvari, ki jih želijo povedati*« (R4). »*Odkritost pri pogovarjanju o izdelku, ki ga razvijamo, vsak lahko predlaga svoje ideje za rešitev, pripadnost skupini, vse to omogoča boljše sodelovanje*« (R2). »*Scrum bolj spodbuja komunikacijo med člani ekipe. To se vidi na dnevnih sestankih, pregledih sprinta in retrospektivah, kjer se ekipa ukvarja tudi sama s seboj in ne samo s projektom. Na dnevnem sestanku smo se dogovorili tudi za morebitne sestanke tekom dneva za posamezno uporabniško zgodbo, kjer so bili udeleženi različni profili tehničnih kadrov, ki so skupaj sodelovali pri razreševanju problemov«* (SM). Boljše sodelovanje razvijalcev pa ni samo po sebi umevno: *»Na začetku projekta je bilo treba to sodelovanje spodbujati, saj je večina prišla iz waterfall projektov, kjer so bili navajeni, da jim projektni vodja delegira nalogo in se niso ukvarjali s celoto, npr. razviti rešitev od baze do frontenda, za kar je potrebno veliko sodelovanja. Postopoma smo dosegli to, da so skupaj iskali rešitve in ne čakali, da jim bo kdo delegiral naloge. To smo dosegli tako, da smo začeli obravnavati uporabniške zgodbe kot celote in ne po posameznik nivojih, npr. baza, backend, frontend, kot je bilo sprva na projektu. V Jiri smo imeli eno nalogo Jira za eno uporabniško zgodbo, ki so si jo razvijalci podajali med seboj, da so skupaj razvili celotno zgodbo. Pred*

tem je bila zgodba razdrobljena na več lističih/nalogah, kjer se je izgubila celotna slika. Na pregledih sprinta smo uvedli igro Best Sprint Task Force, kar je tudi spodbudilo sodelovanje« (SM). Odgovori naših intervjuvancev potrjujejo Overhagejevo in Schlaudererjevo hipotezo. Razvijalci raje delajo skupaj kot posamično in komunikacija pri Scrumu je bolj učinkovita ter takojšnja, zato se tudi člani skupine med seboj bolj povežejo.

Sedma hipoteza: Metodologija Scrum **zmanjša kompleksnost procesa** razvoja programske rešitve kot klasične metodologije. Intervjuji so pokazali, da se večina ne strinja s tem, da Scrum zmanjša kompleksnost procesa razvoja programske rešitve, eden celo trdi, da jo lahko tudi poveča. Na eni strani ga razbitje funkcionalnosti na uporabniške zgodbe naredijo bolj obvladljivega, ne pa tudi manj kompleksnega: »*Scrum kompleksnosti procesa ne zmanjša, ampak ga naredi bolj obvladljivega, ker vso stvar razdeli na manjše cilje, ki jih lažje obvladuješ. Pri tem moraš paziti, da ne zgubiš celotne slike. Od začetka smo imeli težave s celotno sliko in tudi člani skupine so se pritoževali nad tem, a kasneje smo dosegli, da je ekipa imela celotno sliko. To smo dosegli z uporabniškimi zgodbami kot celotami, s pregledi sprintov, drugačnim načinom uporabe Jire, kjer smo uvedli epice in zgodbe združevali po funkcionalnostih v epice (poleti 2014 smo jih začeli uvajati in nekaj časa je trajalo, da smo jih začeli uporabljati)*« (SM). »*Še zmeraj moraš dati vse faze skozi: analiza, testiranje, razvoj in Scrum nič od tega ne izpušča, samo vrstni red je drugačen*« (R2). Na drugi strani, pa nekateri obvladljivost procesa povezujejo s kompleksnostjo: »*Problemi se razdelijo na manjše dele in ti so bolj obvladljivi. Proces je bolj obvladljiv, ker sproti zaznavaš morebitne težave, jih odpravljaš in gradiš dobro rešitev*« (R1). »*Iterativni proces, drobljenje funkcionalnosti na uporabniške zgodbe in prioritiziranje posameznih zgodb olajša delo. Skrbnik produkta lahko sproti prioritizira zgodbe, kar tudi zmanjša kompleksnost*« (R4). En razvijalec meni, da Scrum lahko celo poveča kompleksnost procesa: »*Do neke mere Scrum proces zakomplicira, saj če Scruma ne poznaš, ga moraš spoznati in sprejeti prakse. Če nimaš usposobljene ekipe oz. ni dovzetna za delo po Scrumu, se lahko Scrum izjalovi*« (R4). Ti odgovori se ujemajo z rezultati Overhageja in Schlaudererja, zato te hipoteze ne moremo niti potrditi niti ovreči.

Osma hipoteza: Metodologija Scrum **zahteva več discipline** pri razvoju programske rešitve kot klasične metodologije. Intervjuji so pokazali, da se vsi razen enega vprašanega strinjajo s tem, da Scrum zahteva več discipline od razvijalcev kot tradicionalne metodologije razvoja. Disciplina zanje ni moteča stvar, saj ta ob rednem izvajanju praks preide v navado: »*Na dnevnem sestanku moraš podajati bistvo, sproti beležiti delo v Jiro, se dnevno zavedati, kje pri razvoju si in zavedanje celotne iteracije, da do pregleda sprinta narediš delujoče celote in to po konceptu »done«, vse to je stvar discipline. Tej disciplini sem se prilagodil in mi odgovarja*« (R4). »*Striktno se je treba držati urnika, vsak dan ob istem času na istem mestu za dnevni sestanek, pregledi sprintov in planiranja sprintov. Če želiš po dvotedenskem sprintu res zaključiti, moraš kot razvijalec imeti to odgovornost, da se teh rokov držiš. Ni prostora za kampanjskost, saj imaš vedno stalen ritem. Ko se stvari z disciplino navadiš, npr. deljenje zgodb v Jiri, dnevni sestanki, ti to pride v navado*« (SM). »*Delati moraš sproti, saj je na vsakih 14 dni pregled sprinta, ko morajo biti zgodbe izdelane. Tudi pri Jiri moraš biti*

dosleden in sproti posodabljati status uporabniške zgodbe (to do, in progress, in testing, done) […], da sam testiraš stvari, ki si jih razvil, preden daš zgodbo v testiranje testerju« (R1). Ena oseba pa se ne strinja s tem, da Scrum zahteva več discipline: »*Scrum omogoča to, da se nekdo bolj vključi v skupino in zaradi tega dobi večjo motivacijo, da nekaj naredi*« (R3). Naši intervjuvanci so potrdili hipotezo Overhageja in Schlaudererja. Za razliko od njunih ugotovitev, ki sta disciplino intervjuvancev, ki jo Scrum zahteva, zaznala kot pomanjkljivost, ki negativno vpliva na sprejetje Scruma, so izsledki našega intervjuja pokazali, da je disciplina pri izvajanju praks pomembna, a ji je treba dati čas, da razvijalcem prakse preidejo v navado in tako lahko v polni meri izkoristijo prednosti metodologije Scrum napram tradicionalnim metodologijam razvoja programske opreme.

Prepoznali smo več faktorjev, ki delujejo kot motivatorji pri sprejemanju metodologije Scrum, pa naj bo to kot relativna prednost ali kot večja združljivost z dejanskimi delovnimi praksami v primerjavi s tradicionalnimi metodologijami razvoja programske opreme. Kljub temu pa smo naleteli tudi na morebitne ovire za sprejem Scruma, saj sta faktorja značilna za kompleksnost, zaznana različno. Nekdo celo vidi večjo kompleksnost procesa kot oviro za uporabo Scruma, kar spet potrjuje, da je Scrum lažje razumeti kot ga izvajati v praksi. Ker je zaznavanje teh faktorjev v nasprotju z enim od osrednjih ciljev metodologije Scrum, lahko predstavljajo nevarnost za trajnost uporabe Scruma pri razvoju programskih rešitev.

7.4 REZULTATI SPECIFIČNIH VPRAŠANJ NA PODLAGI ZNAČILNOSTI PROJEKTA

V nadaljevanju bomo predstavili rezultate intervjujev z izbranimi tipičnimi predstavniki skupine Scrum, ki so prispevali svoje poglede in stališča glede značilnosti projekta, kar je predmet raziskovanja RC4.

Kot je bilo že omenjeno, je imela skupina na projektu v začetku težave s **konceptom »done«**. Ker je ta koncept za kakovost in hitrost razvoja zelo pomemben, ga je smiselno natančneje preučiti. Vsi vprašani so se strinjali, da so bile na začetku projekta težave pri sledenju temu konceptu: »*Nismo dosledno pisali dokumentacije in testirali pred pregledi sprinta*« (R1). »*Na začetku smo imeli težave, ker smo se Scrum še učili. Ob koncu smo se temu konceptu bolj približali*« (R3). »*Sprva smo pri planiranju sprinta posebej ocenjevali razvoj in testiranje, ne pa zgodb v celoti, kot je po konceptu done*« (R4), »*Učinki nepravilno delujoče kode niso nujno vidni takoj, ampak šele čez čas, zato mogoče nismo bili toliko dosledni pri izvajanju koncepta done*« (R2). Ker je skupina zaznala, da ima težave s sledenjem konceptu »*done*«, je uvedla spremembe, ki so izboljšale njegovo uporabo: »*Uvedli smo pravilo, da je pred pregledom sprinta potrebno testirati uporabniške zgodbe in smo temu namenili tudi čas, česar prej nismo imeli. Od takrat smo tudi pri oceni uporabniške zgodbe upoštevali čas testiranja, ne samo čas razvoja*« (R1). »*Težave smo rešili tako, da je SM3 stalno spodbujal razvijalce, da delajo zgodbe skupaj, ne samo vsak svoj košček, torej da so skupaj odgovorni za zgodbe.*

Razvijalci so začeli svoje izdelke sami predstavljati na pregledih sprintov in jim je bilo v interesu, da prikažejo delujočo zgodbo in dokumentirali uporabniška navodila ter navodila za namestitev« (SM). Zaradi časovne stiske proti koncu projekta pa je bil koncept »done« prilagojen takratni situaciji: »*Proti koncu projekta nismo bili ažurni pri uporabniških navodilih zaradi pomanjkanja časa, so pa bila ažurna navodila za nameščanje, saj so bila v tistem času najbolj pomembna za produkcijo*« (SM). »*Zaradi časovne stiske je bila definicija »done« prilagojena, npr. brez unit testov*« (R4).

Na projektu so se razvrstili kar trije **skrbniki metodologije**, kar je lahko dobro, če gre razvoj po metodologiji Scrum zaradi tega na bolje. Glede na to, da je bil to za podjetje prvi projekt po metodologiji Scrum in da večina članov skupine ni imela praktičnih izkušenj z njim, nas je zanimalo je, ali se jim zdi vloga skrbnika metodologije pomembna. Vsi so se strinjali s tem, da je ta vloga pomembna, ker: »*SM nas usmerja, kadar skupina zaide. Največ problemov smo imeli na dnevnih sestankih, saj smo velikokrat začeli razglabljati o rešitvah problemov, ne pa o tem kaj smo in bomo delali ter katere probleme imamo. Tudi pri planiranju je SM pomemben, da se diskusija za posamezno zgodbo ne razvleče v neskončno debato. SM skrbi za učinkovitost skupine*« (R1). »*Na začetku je njegova vloga zelo pomembna, saj je dobro, da skrbi za red in disciplino, dokler skupina teh pravil ne ponotranji in preidejo v navado. Da nas opozarja na časovne omejitve in smernice metodologije*« (R2). »*Zaradi fokusa skupine, da sledi disciplini in praksam Scruma. SM je odgovoren za učinkovitost izvajanja metodologije in posledično so tudi rezultati boljši*« (R4). »*Ko vpeljuješ Scrum je ta vloga zelo pomembna, saj moraš ekipo učiti o Scrumu, ji razlagat,i zakaj je Scrum dober, opazovati, kaj gre ekipi dobro in kaj ne, zaznati, kdaj je kakšna praksa prehitro vpeljana in poiskati drug način. Potrebno je brati literaturo na to temo. Na začetku potrebuješ veliko energije*« (SM). Vsi ti odgovori kažejo, da je vloga SM pri vpeljavi Scruma zelo pomembna, medtem ko je njegova vloga pri že vpeljani skupini precej manjša: »*Če je ekipa uigrana, je ScrumMaster v vlogi opazovalca, ki samo občasno korigira*« (SM).

Pri samem **sodelovanju stranke**, so si bili intervjuvanci enotni, da bi lahko bila bolj vključena v proces razvoja rešitve, predvsem s testiranjem in posredovanjem sprotne povratne informacije: »*je bila premalo udeležena na začetku projekta in ni podajala povratne informacije, zato smo razvijali stvari, ki jih ni potrebovala. Želel bi si, da bi sproti testirala verzijo izdelka, ki smo ji jo dostavili. Dosledno so začeli testirati šele po produkciji*« (R2). »*Nekatere funkcionalnosti bi lahko bolj natančno definirali, sproti testirali in dajali povratno informacijo*« (R1). Razvijalci so zaznali, da bi bil produkt lahko še boljši, če bi stranka več sodelovala: »*Stranka ni bila dovolj vpletena v razvoj rešitve in ni dobila tistega, kar je potrebovala, ampak tisto, kar smo mi predvidevali, da potrebuje*« (R3). »*Stranka bi lahko vplivala na izboljšanje produkta. Zavzeti bi morala vlogo produktnega vodja, saj je naš PV1 po lastni presoji določal prioritete glede na PZI*« (R4). »*Proti koncu so začeli dajati povratno informacijo, ki bi jo lahko dali že prej, in bi kakšno stvar še bolje naredili*« (SM). Da bi se v prihodnje izognili takim težavam, je treba poiskati vzroke za nezadostno sodelovanje stranke: »*Bila je navajena na waterfall in datum produkcije ji je bil predaleč. Prej bi jo morali*

vključiti v načrtovanje in pregled sprinta. Ko smo jo proti koncu projekta vključili, ji mogoče nismo na pravi način razložili, kaj je njena vloga pri načrtovanju sprinta, saj ni razumela, da ni ona tista, ki vodi sestanek, ampak je tam zato, da daje odgovore na naša vprašanja. Po drugi strani je bilo dobro, da smo iz prve roke dobili informacije o njenih prioritetah in včasih se je glede na visoko oceno zgodbe odločila, da nečesa ne potrebuje, saj je videla, da terja preveč časa« (SM). »*Stranka ni dovolj naredila za razvoj produkta, ker je šele pol leta pred produkcijo začela testirati rešitev, saj so si šele takrat vzpostavili testno okolje. Takrat je začela razmišljati, ali ji rešitev odgovarja ali ne, čeprav so jim bile verzije dostavljene že prej. Šele takrat je prišla povratna informacija o izdelku z njihove strani*« (R4). »*Preden so si postavili svoje testno okolje, so imeli možnost testirati izdelek na našem okolju, a te možnosti niso uporabili toliko kot bi jo lahko*« (SM). Kljub temu so se vsi intervjuvanci strinjali, da je stranka na koncu bila zadovoljna z izdelano rešitvijo.

Proti koncu projekta je bila podpora metodologiji Scrum v fizični (tabla Kanban) in elektronski (Jira) obliki, zato nas je zanimalo, če je bila fizična **tabla Kanban** razvijalcem odveč ali dobrodošla. Glede tega so bila mnenja razvijalcev deljena. Večina se strinja, da je fizični Kanban dobrodošel, saj omogoča celostni pogled na sprint: »*Ima dober učinek, če si dosleden pri njenem ažuriranju. Na hitro vidiš celotno sliko sprinta oziroma, kje so ozka grla. Všeč so mi tudi barve listkov za posamezne vrste nalog, na primer oranžna za napake, zelena za izboljšave, rumena za uporabniške zgodbe itd.*« (R2). »*Celotna slika sprinta je bolj vidna, kar je še posebej dobrodošlo pri dnevnem sestanku. Občasno smo pozabili na ažuriranje te table*« (R1). »*Je pomembna, ker lahko ekipa stalno vizualizira napredek in stanje dela. Fokus na tabli je zjutraj na dnevnem sestanku, po potrebi pa še vmes. V Jiri se celotna slika sprinta izgubi, saj je monitor premajhen*« (R4). »*Je bolj transparentna kot Jira, saj vsakič, ko greš mimo vidiš, kje v sprintu smo. Če naloge nekje stojijo, skušaš najti razloge za to in jih odpraviti, da se stvari premaknejo naprej*« (SM). Eni osebi pa zadostuje samo Jira: »*Ne zdi se mi pomembna, saj mi jemlje fokus in koncentracijo. Jira mi zadostuje*« (R3). Rezultati intervjujev potrjujejo rezultate vprašalnika, kjer je bila fizična tabla Kanban poleg elektronske dobro sprejeta.

Ocenjevanje uporabniških zgodb je sprva potekalo v urah, pozneje v točkah. Intervjuvance smo spraševali, ali se jim zdi **ocenjevanje v točkah bolj objektivno kot v urah**. Odgovori so bili mešani, eni se strinjajo, da je lažje ocenjevati v točkah: »*V točkah je lažje, saj podaš grobo oceno zgodbe in se ni potrebno ukvarjati z manjšimi niansami pri sami oceni, kot pri ocenjevanju v urah. Pri točkah ni potrebno razbijati ocene po posameznih kosih, npr. razvoj, testiranje, dokumentiranje, ampak gledaš zgodbo kot celoto*« (R4). »*Da, saj je nekdo lahko hitrejši kot drugi, zahtevnost zgodbe ostaja enaka. Menim, da smo že po 2. ali 3. planiranju s Planning Pokrom znali dobro oceniti težavnost posameznih uporabniških zgodb*« (R1). Eni trdijo, da ocene v urah niso realne, saj je en razvijalec pri razvoju zgodbe hitrejši, drug pa počasnejši: »*Opazila sem, da ko smo ocenjevali zgodbe v urah, je vsak član skupine ocenjeval, koliko ur bi za to zgodbo porabil. Eni so bolj hitri, drugi bolj počasni in take ocene niso realne. Pri točkah se npr. neka zgodba zdi kompleksna za 3 točke in jo primerjaš z neko*

drugo zgodbo za 3 točke, skratka primerjaš kompleksnost zgodb. Za tako zgodbo lahko nekdo porabi 1 teden, drugi 3 ure« (SM). Na drugi strani pa nekateri trdijo, da pri ocenjevanju v točkah na nek način pretvoriš predvidene porabljene ure: »*Težavnost zgodbe je lahko minimalna, a lahko zanjo porabiš veliko časa, zato se mi ocenjevanje v točkah ne zdi smiselno, saj na drug način predstaviš čas porabe za izdelovanje posamezne zgodbe […] bolje bi bilo, da bi ocenjevali v urah in izračunali povprečje ur, ki so jih razvijalci predlagali za posamezno zgodbo*« (R3). Posamezniki imajo deljena mnenja o načinu ocenjevanja uporabniških zgodb, je pa to pomembno za načrtovanje sprinta, da lahko določiš velikost seznama zahtev sprinta. Mogoče bi skupina potrebovala več vaje ali pa drugo igro za ocenjevanje uporabniških zgodb: »*Od začetka bi mogoče bila igra Team Estimation boljša, da bi dobili občutek, kakšna je velikost zgodbe S, M ali L. Ko razvrščaš zgodbe na tabli, vidiš, kaj si dal na isti kup in tako lažje primerjaš velikost zgodb ter dobiš celotno sliko sprinta*« (SM).

Glede na začetne težave pri sprejemanju praks Scrum na projektu nas je zanimalo, kako razvijalci vidijo **uvedbo praks na projekt Scrum**, in sicer vse naenkrat ali postopoma. Mnenja razvijalcev so različna, eni zagovarjajo postopno vpeljavo praks: »*Postopoma, da ni toliko novih stvari in se lahko posvetiš tudi razvoju rešitve*« (R1). »*Boljše postopoma, če skupina še nima izkušenj s Scrumom, saj je preveč informacij naenkrat. Če pa ekipa prej izvede npr. 1-tedensko delavnico na temo razvoja po Scrumu na fiktivnem primeru, potem bi bilo lažje uvesti vse naenkrat*« (R2). Drugi zagovarjajo uvedbo vseh praks naenkrat: »*Boljše je uvesti vse prakse naenkrat in jih potem prilagajati svojim potrebam. Skupina mora skozi evolucijo, da zazna pozitivne učinke metodologije Scrum in kako lahko to uporabijo sebi v prid. Uvajanje metodologije Scrum mora potekati po metodologiji Scrum – dobiš projekt in skozi iteracije prilagajaš produkt; prilagajanje metodologije se izvaja s pomočjo retrospektive*« (R4). »*Literatura priporoča, da vse prakse uvedeš na začetku in šele, ko razumeš njihov pomen, jih lahko prilagajaš. SM mora biti pozoren ali je katera praksa prezgodaj uvedena. Odvisno je tudi od ljudi v ekipi, koliko so naklonjeni Scrumu. Pri članih, ki prihajajo iz projektov waterfall, je vpeljava Scruma bolj dolgotrajen proces*« (SM).

7.5 REZULTATI POSTMORTEM ANALIZE

Za konec smo v duhu postmortem analize pri intervjuvancih preverjali negativne in pozitivne strani projekta s stališča metodologije Scrum, ki se tudi navezuje na RC4.

Če začnemo z **negativnimi** stranmi projekta s stališča metodologije Scrum, so intervjuvanci največkrat izpostavili premajhno sodelovanje stranke: »*Na našem projektu je bil naročnik premalo vpleten v razvoj rešitve, lahko bi sproti testirati verzije izdelka in bili prisotni na pregledih sprintov*« (R1). »*Težava na našem projektu je bila vpetost naročnika v razvoj. Naročnik bi moral biti že od začetka projekta zraven, saj dolgo časa ni dajal povratne*

informacije. Da bi se udeleževal groominga in nam odgovarjal na naša vprašanja in s tem razjasnjeval zadeve. Naročnika v vlogi produktnega vodja ne vidim, to mora biti sodelavec iz podjetja« (R3).

Poleg premajhnega sodelovanja stranke so intervjuvanci omenjali tudi nezainteresiranost posameznikov za Scrum v razvojni skupini: »*Imeli smo tudi nekaj članov, ki takemu načinu razvoja niso bili naklonjeni, zato je pri izbiri članov treba skrbno izbirati take, ki so pripravljeni delati agilno*« (R1). »*Če ljudje niso pripravljeni razvijati po Scrumu, potem imaš težave*« (SM). »*Če član ekipe ni pripravljen sodelovati, ga skupina sama izloči*« (R2).

Kot pomanjkljivost je bila izpostavljena velikost razvojne skupine: »*Mogoče nas je bilo preveč v skupini, to je več kot 10 ljudi. Zaradi tega je bilo težko slediti vsem zgodbam na dnevnih sestankih, ker jih je bilo preveč. Rešitev za to bi bilo manj ljudi na projektu. PV1 je želel razvijati več funkcionalnosti vzporedno in je zato potreboval več ljudi, ampak manj ljudi bi to hitreje naredilo, saj bi bilo manj usklajevanja. Ne bi delali 5 stvari hkrati, ampak 3 in te bi bile dobro narejene*« (SM). Prevelika razvojna skupina na obravnavanem projektu tudi ni v skladu s priporočili Scruma, ki priporoča od 3 do 9 razvijalcev na razvojno skupino [23]. Da je skupina prevelika, se je omenjalo tudi pri ocenjevanju zgodb: »*ekipa je velika in vsi ne poznajo zahtevnosti posamezne zgodbe, zato pri oceni ugibajo in bi bilo bolje, če taki ne bi ocenjevali, saj prihaja do napačnih ocen*« (R2).

Nekatere je zmotila neizkušenost prvih dveh skrbnikov metodologije: »*Prva dva SM nista bila dovolj usposobljena za Scrum in s premalo entuziazma. Bili smo začetniki pri Scrumu in smo potrebovali veliko usmerjanja s strani SM. Pri novi ekipi mora biti SM skrbno izbran, da zna usmerjati skupino*« (R4), kar je razumljivo, saj se je večina članov skupine Scrum prvič srečala s razvojem konkretne rešitve po metodologiji Scrum.

Za zaključek pa izpostavimo **pozitivne** strani projekta s stališča metodologije Scrum. Največkrat je bilo omenjeno skupinsko delo, komunikacija skupine in povezanost članov skupine: »*Zaupanje v skupini je bilo veliko, tako da si lahko na retrospektivi sprinta brez zadržkov povedal, kaj ti je glede procesa razvoja všeč in kaj ne. Igra jadrnica je fina, ker ves čas visi na steni, in te spominja, kaj je bilo predlagano. Na naslednji retrospektivi smo z njeno pomočjo hitro preverili ali smo upoštevali predloge zadnje retrospektive, saj se niso predlogi 'izgubili' nekje v Confluenceu*« (R1). »*Delo v skupini, saj spodbuja komunikacijo*« (R2). »*Komunikacija med člani skupine in vpetost v skupino. Če čutiš pripadnost projektu/skupini, boš tudi več sebe vložil v projekt in stremel k dobrim rezultatom. Pri waterfallu tega občutka pripadnosti ni toliko kot pri Scrumu*« (R3). Ta povezanost skupine ob koncu projekta pa ni nastala sama od sebe, ampak je bila posledica dolgotrajnega procesa sprejemanja Scruma in drugačnega načina dela, kar je bilo za večino težko, saj je večina posameznikov prihajala iz tradicionalnih projektov. Naj spomnimo, vmes je bila skupina na celodnevni delavnici komuniciranja, tako slaba je bila komunikacija med člani skupine. Da je bila skupina na koncu zelo povezana, kažejo tudi rezultati iger, ki jih je skupina izvedla proti koncu projekta: »*Scrum je ustvaril ekipo, ki je dokaj samostojna in samoorganizirana. V tej skupini so ljudje,*

ki jim takšen način dela ustreza in so sprejeli metodologijo Scrum. Vzdušje je bilo dobro, kar je pokazala tudi igra Team Barometer, ki smo jo izvedli proti koncu projekta in je imela zelo dobre rezultate. Tudi igro Agile Wheel smo izvedli dvakrat v razmiku 6 mesecev – osebno zadovoljstvo je bilo v drugem krogu boljše ocenjeno kot prvič« (SM).

Krajše iteracije so posamezniki izpostavili kot prednost, saj prinašajo hitrejše rezultate in večje zadovoljstvo: *»Krajše iteracije omogočajo večkratni pregled izdelanega, kar prinaša konstantno zadovoljstvo*« (R1). »*Razvoj v sprintih pospeši razvoj programske opreme in da ekipi motivacijo, saj smo ponosni na delujočo rešitev. Ko je PV1 predstavljal izdelke, tako velikega zadovoljstva ni bilo, kot takrat, ko smo zgodbe na pregledih sprinta predstavljali sami*« (R4). »*Kratka časovna omejitev sprintov ti da motivacijo, da stvari zaključuješ in ne dopušča kampanjskosti*« (R2).

Scrum prispeva tudi k boljši samopodobi posameznika: »*Na tem projektu sem se naučil vprašati za nasvet in vsebinsko razlago. Na začetku mi je bilo na pregledih sprinta težko pred skupino predstaviti izdelek, ker nisem bil vajen nastopati in nisem bil samozavesten glede svojega dela, saj je to moja prva zaposlitev*« (R1).

Neuspehi so dobrodošli, saj se iz njih lahko veliko naučimo: »*Scrum sam po sebi ne omogoča uspeha, saj striktno izvajanje praks ne prinaša napredka, ampak prej nezadovoljstvo, saj v teh praksah ni pravega smisla. To se je pokazalo tudi na našem projektu, kjer je bilo nekaj trenutkov, ko smo se vprašali, ali naj s Scrumom nadaljujemo. Po ukinitvi vseh praks in ponovnem postopnem uvajanju praks, ki so začele dobivati smisel, se je napredek ekipe povečal, tudi odnosi v ekipi so se izboljšali, kar je zagotavljalo boljše sledenje konceptu »done« in s tem kvaliteti rešitve. Z znanjem, ki smo si ga pridobili na tem projektu nam bo še kako koristilo pri prihodnjem delu, saj so nam začetni neuspehi z metodologijo Scrum dali pomembne lekcije.*« (SM).

8 PRIPOROČILA ZA NADALJNJO UPORABO METODOLOGIJE SCRUM

Veliko podjetij po svetu se trudi postati čim bolj agilnih in zato se odločijo za uvedbo ene izmed agilnih metodologij, največkrat izberejo metodologijo Scrum [32]. Njihove skupine pričnejo delati v iteracijah, vsako jutro imajo dnevni sestanek in pričnejo lepiti listke po stenah. Izkaže se, da zgolj drugačen način dela ne prinese želenih rezultatov in višje produktivnosti, saj je predpostavka, da če uporabljaš neko agilno metodologijo, potem si tudi agilen, napačna. Med »delati agilno« in »biti agilen« je namreč velika razlika, saj prvo pomeni zgolj slediti določenim praksam, drugo pa resnično delovati v skladu z agilno miselnostjo [8].

Pri priporočilih za nadaljnjo uporabo Scruma smo se osredotočili tako na stvari, ki jih je potrebno izboljšati, kot tudi na tiste, ki so se med delom na projektu izkazale kot pozitivne in jih je smiselno obdržati, da bo Scrum dolgoročno sprejet.

Priporočila se nanašajo na raziskovalni cilj RC4.

8.1 PRIPOROČILA ZA IZBOLJŠANJE UPORABE METODOLOGIJE SCRUM

Rezultati analiziranega projekta so pokazali nekatere pomanjkljivosti pri uporabi metodologije Scrum in njenih priporočil, ki se nanašajo na skrbnost izbire skupine Scrum, vključitev naročnika v proces Scruma, sprotno spremljanje poteka dela in ocenjevanje uporabniških zgodb.

8.1.1 SKRBNA IZBIRA SKUPINE SCRUM

Zanimanje posameznikov za novost, ki se vpeljuje, je ključno za uspeh projekta, zato je skrbna izbira članov skupine Scrum zelo pomembna, še posebej oseb za produktnega vodja in skrbnika metodologije. Da sta ti dve vlogi pomembni, so pokazali tudi rezultati vprašalnika, saj so vprašani ocenili, da sta usposobljenost in osebna angažiranost skrbnika metodologije in produktnega vodja ključni za uvajanje metodologije Scrum, pri čemer je bilo povprečje odgovorov 6,60 (SM) in 6,27 (PV) od 7 možnih točk. Na obravnavnem projektu sta se zvrstila dva produktna vodja in trije skrbniki metodologije. Ankete in intervjuji so pokazali, da so bili

razvijalci večinoma naklonjeni metodologiji Scrum, medtem ko je bil PV1 tej metodologiji nenaklonjen, kar se je kazalo tudi v velikih težavah pri vpeljavi metodologije Scrum, še posebej v prvi polovici projekta. Tudi izbor prvega SM ni bil najbolj ustrezen, saj se SM1 v tej vlogi ni videl in zato tudi ni posvečal dovolj časa in pomembnosti vlogi skrbnika metodologije. Naslednji skrbnik metodologije, to je SM2, je poznal prakse in priporočila Scruma, a mu skupina ni sledila, saj je ni uspel prepričati v smiselnost izvajanja predpisanih praks. Rezultati intervjujev kažejo, da je vloga skrbnika metodologije na prvem projektu oz. z ljudmi, ki še niso delali po Scrumu, ključnega pomena. Šele tretji skrbnik metodologije, to je SM3, je postopoma vpeljal prakse Scrum in skupini omogočil, da je v njih videla smisel in čutila potrebo po njih.

Intervjuji so pokazali, da so člani skupine zelo pomembni, ali bo Scrum sprejet ali ne. Na projektu prenove portala za državljane in zalednih sistemov je bilo nekaj ljudi, ki so že delali na agilen način ali pa je bila to njihova prva izkušnja razvoja programske rešitve na konkretnem realnem primeru, kar je vsekakor dobro vplivalo na sprejetost metodologije Scrum, saj niso bili ukalupljeni na tradicionalni način dela in razmišljanja. Precej oseb je dolga leta delalo po slapovni metodologiji in ti so imeli kar nekaj težav pri miselnem preskoku in drugačnem načinu dela. Tudi med člani razvojne skupine so bili nekateri nezainteresirani za Scrum, a jih je skupina sama izločila.

Pri oblikovanju skupine Scrum je treba paziti tudi na velikost skupine in se držati priporočil Scruma, to je od 3 do 9 oseb [23]. Prevelika skupina je lahko neučinkovita, saj posameznik težko sledi vsem uporabniškim zgodbam in jih težko ocenjuje, ker jih ne pozna dovolj dobro, kar je pokazal tudi naš intervju.

8.1.2 VKLJUČITEV NAROČNIKA

Vključitev naročnika je zelo pomembna, saj le naročnik lahko poda vizijo projekta in ustrezno povratno informacijo, ki je osnova za smer razvoja v prihodnje. Naročniku je bilo ob začetku obravnavanega projekta omenjeno, da bo razvoj tekel po metodologiji Scrum. Glede na to, da se je skupina Scrum metodologiji še privajala in imela začetne težave, tudi naročnika ni vključevala zraven, ampak mu je le dostavljala kodo ob dogovorjenih mejnikih.

Med samim projektom se je na strani naročnika popolnoma zamenjala skupina, ki je odločala o vsebini in prioritetah pa tudi svoje testno okolje so postavili zelo pozno. Naročnik je imel možnost testiranja na izvajalčevem okolju, česar pa ni v celoti izkoristil. V sam proces Scruma je bil vključen relativno pozno, saj ga je šele PV2 povabil na načrtovanja in preglede sprintov. Pozna vključitev je pomanjkljivost in tudi v nasprotju s priporočili Scruma, kar kažejo tudi rezultati intervjujev. Intervjuvanci omenjajo, da bi si želeli zgodnejše vključenosti naročnika, da bi sproti testiral izdelke in jim sporočal povratno informacijo, saj si želijo biti še boljši in naročniku narediti izdelek, ki jim bo ustrezal, ne pa nujno tistega, ki je bil prvotno definiran. S tem bi povečali kakovost rešitve, zadovoljstvo razvijalcev kot tudi naročnika.

Smiselno bi bilo naročniku na začetku projekta podrobneje predstaviti potek Scruma, vloge in prakse, pri tem pa jasno določiti njegovo mesto v Scrumu in mu povedati, kaj skupina Scrum od njega pričakuje.

8.1.3 SPROTNO SPREMLJANJE POTEKA DELA

Sprotno spremljanje poteka dela je pomembno za sledenje planom, pri odpravljanju morebitnih zaznanih težav in določanju hitrosti skupine. Vprašalniki so pokazali, da praksa diagram Burndown na projektu ni bila dobro sprejeta, saj je dosegla le 2. stopnjo sprejetja. Intervjuji pa so razkrili vzroke zanje, to je nedosledno vnašanje ur in demotivacija razvojne skupine ob prepočasnem spuščanju diagrama. Mnenja intervjuvancev glede smotrnosti spremljanja poteka dela s pomočjo diagramov so bila deljena, a za boljši nadzor nad potekom dela bi jih bilo smiselno vpeljati in predstaviti njihovo pomembnost skupini Scrum. V pisarni bi bilo treba najti mesto, kjer bi bili ti diagrami vidni ves čas in na tak način približali spremljanje poteka dela skupini Scrum.

8.1.4 OCENJEVANJE UPORABNIŠKIH ZGODB

Ocenjevanje uporabniških zgodb je pomembno za načrtovanje sprintov in tega se zaveda tudi skupina Scrum, kar so pokazali rezultati vprašalnikov. Igra *Planning Poker* je dobro sprejeta praksa, medtem ko je *Team Estimation Game* slabo sprejeta, kar je razumljivo, saj predstavlja alternativo *Planning Poker*ja. Intervjuji so razkrili, da se skupina Scrum ni povsem poenotila oziroma našla skupnega imenovalca glede merske enote ocene uporabniške zgodbe. Eni bi raje ocenjevali v točkah, drugi v urah. Čeprav se je proti koncu projekta ocenjevalo v točkah, SM3 opaža, da vsi člani skupine nimajo enakega pogleda na točkovanje uporabniških zgodb in zato te ocene niso preveč zanesljive oziroma prihaja do večjih odstopanj.

Za boljše planiranje sprintov in ocenjevanje hitrosti skupine bi bilo treba izboljšati ocenjevanje uporabniških zgodb. V ta namen predlagamo uporabo matrike napora [31], pri kateri število točk za vsako uporabniško zgodbo določimo na podlagi dveh kriterijev: velikosti in kompleksnosti. Velikost zgodbe je relativna ocena obsega dela v smislu dejanskega napora za razvoj. Primer kriterijev zgodbe velikosti 1:

- zelo majhna zgodba, ki predstavlja majhno stopnjo napora,
- potrebnih je le nekaj ur dela.

Kompleksnost se nanaša na zahteve zgodbe in/ali tehnično izvedbo le-te. Kompleksnost predstavlja negotovost ocene, to je večja kompleksnost pomeni večjo negotovost. Primer kriterijev zgodbe kompleksnosti 1:

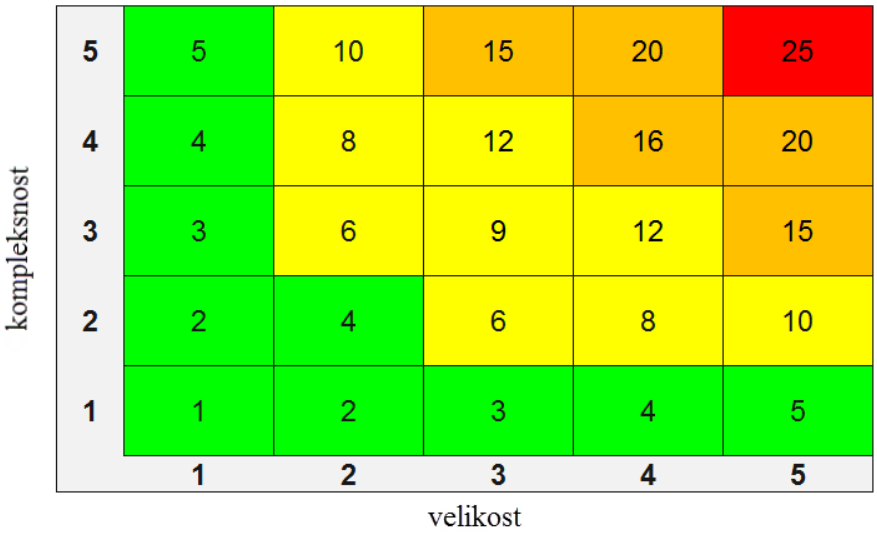
- zelo preprosta, z majhnim številom neznank,
- tehnične in poslovne zahteve so zelo jasne in brez dvoumnosti,
- dodatne raziskave niso potrebne,

- zahtevano je osnovno znanje programiranja za izdelavo zgodbe.

Razvojna skupina si mora postaviti svoja merila za določanje velikosti in kompleksnosti zgodb.

Z uporabo teh dveh kriterijev, kot prikazuje Slika 13, se napor določi s preprosto formulo:

napor = velikost x kompleksnost.



Slika 13: Matrika napora

8.2 PRIPOROČILA ZA OHRANJANJE UPORABE METODOLOGIJE SCRUM

Rezultati vprašalnikov in intervjujev so pokazali tudi pozitivne vidike Scruma, ki jih je smiselno ohraniti, saj prispevajo k boljši sprejemljivosti Scruma. Ti vidiki se nanašajo na skupne prostore in sodobno infrastrukturo, osebno in strokovno rast posameznika ter možnost dela po metodologiji Scrum tudi v prihodnje.

8.2.1 SKUPNI PROSTOR IN INFRASTRUKTURA

Skupina je sprejela Scrum kot inovacijo, saj je korenito spremenila način dela in razmišljanja večine. Intervjuji so pokazali, da ji je bil pri tem v veliko pomoč skupni prostor, kjer so skupaj delali in nemoteno komunicirali med seboj. S tem niso motili drugih razvojnih skupin v podjetju, kar je zagotovo osnova za **sproščeno in neformalno komunikacijo**. Vprašalniki in intervjuji so pokazali, da se je prav neformalna komunikacija izkazala za zelo pomembno pri doseganju samoorganiziranosti, občutka pripadnosti skupini in večje motivacije posameznikov pri razvoju programske rešitve.

Skupna soba omogoča tudi **transparentnost**, kjer ažurna fizična tabla Kanban ponuja celotno sliko sprinta, kar se v Jiri pogosto izgubi zaradi omejenega prikaza.

Člani skupine so imeli na voljo tudi **vso potrebno tehnološko infrastrukturo**, od Jire za upravljanje Scruma kot tudi Jenkins, SVN, Confluence in tako dalje, kar tudi olajša iterativen in inkrementalen razvoj po metodologiji Scrum.

8.2.2 OSEBNA IN STROKOVNA RAST POSAMEZNIKA

Posamezniku je treba tudi v prihodnje omogočiti, da osebno in strokovno raste. Prav tako je treba pustiti razvojni skupini, da sama rešuje probleme, saj so vprašalniki in intervjuji pokazali, da je to razvijalcem v izziv, ker lahko pokažejo svojo kreativnost in znanje, rešitev problema pa jih navdaja z **zadovoljstvom** in **ponosom,** pri tem pa prevzemajo **odgovornost** zanjo. Posameznik mora slediti rutini sprintov in konstantno izdelovati delujočo programsko kodo ter jo na pregledu sprinta predstaviti skupini Scrum, kar tudi prinaša konstantno zadovoljstvo in pozitivno vpliva na njegovo **samopodobo**. Osebe, ki niso vajene komunicirati in nastopati, imajo pri Scrumu priložnost, da izboljšajo svoje **mehke veščine**.

V skupini Scrum mora posameznik poznati tudi druga področja in ne le tisto, na katerem je specialist, če želi komunicirati z drugimi in izdelati ustrezno rešitev, saj ne dobi klasične podrobne specifikacije zahtev kot pri slapovni metodologiji. Pri tem pride do **prenosa znanj** in ob odhodu posameznikov ne ostane velika praznina za njimi, saj se njihovo delo in znanje hitreje in lažje nadomesti. Slednje se je izkazalo tudi na tem projektu, saj se je zamenjalo precej ljudi, a večjih vsebinskih in tehničnih zagat zaradi tega ni bilo. S tem, ko imajo razvijalci širše znanje, se tudi bolj zavzamejo za razreševanje težav drugih, saj področje bolje poznajo.

Intervjuji so pokazali, da Scrum od posameznika zahteva več **discipline**, kar je lahko na začetku ovira. Ob vztrajanju izvajanja praks Scrum postanejo le-te rutina in takrat lahko začne skupina Scrum izkoriščati prednosti metodologije, to so hitrejše izdelani in kakovostnejši izdelki, ki ustrezajo naročniku. Scrum omogoča stalen ritem, kjer ni prostora za »kampanjskost«.

8.2.3 OMOGOČITI NADALJNJE DELO PO METODOLOGIJI SCRUM

Po mnenju anketirancev in intervjuvancev je metodologija Scrum **boljša** in **naprednejša** kot tradicionalne metodologije ter se sklada z njihovim želenim načinom dela. Pri tem so izpostavili **svobodo** pri implementaciji zahtev, ki so seveda v časovnih in vsebinskih okvirih; **pomoč skupine**, ki je neprecenljiva, ko naletiš na problem, ki ga sam ne znaš rešiti; **olajša izvedbo zahtevnejših nalog**, saj so zahteve razdeljene na manjše in obvladljive uporabniške zgodbe.

Intervjuji so pokazali, da je **transparentnost**, ki jo zagovarja Scrum, posameznikom zelo pomembna, saj s tem najdejo tudi svoj prostor v skupini in imajo možnost pomagati drugim pri reševanju težav.

Vprašalnik je pokazal, da si vsi anketiranci želijo, da bi tudi v prihodnje razvijali rešitve po metodologiji Scrum, saj tak način dela prinaša koristi za posameznika, naročnika in ne nazadnje tudi za podjetje. Uvajanje razvoja po metodologiji Scrum pa ne prinese prednosti Scruma čez noč, saj je to proces, ki terja čas in napor.

Za konec pa še misel SM3: *»Ob vsem tem pa ne smemo pozabiti na odnose v skupini, prav njim moramo posvetiti veliko pozornost, saj dobra klima v skupini omogoča, da je skupina samoorganzirana in samoinicativna, da je komunikacija med člani sproščena, da so si člani pripravljeni prisluhniti in si pomagati pri implementiranju nalog, saj je odgovornost za izdelke **skupna**, kar je bistvena razlika v primerjavi s slapovno metodologijo*« (SM), kar potrjuje ugotovitev obsežne raziskave Ceschija in drugih [2], da večina projektov propade zaradi ljudi in posledic projektnega vodenja, ne pa tehničnih izzivov. Prav na te probleme se osredotoča Scrum.

9 SKLEPNE UGOTOVITVE

V magistrskem delu smo analizirali zaključen projekt iz prakse, izdelan po metodologiji Scrum. V podjetju, ki je izvajalo projekt, metodologija Scrum še ni široko uporabljena, zato je analiza dolgoročne sprejemljivosti Scruma zelo pomembna, saj je razvoj po metodologiji Scrum danes dejstvo, če podjetje želi ohranjati konkurenčno prednost. Poleg tega študij o dolgoročni sprejemljivosti Scruma v strokovni literaturi ni veliko in je toliko bolj pomembno, da se osredotočimo tudi na ta segment in dodamo svoj prispevek.

Da smo lahko analizirali faktorje, ki prispevajo k dolgoročni sprejemljivosti metodologije Scrum, smo se oprli na modele iz strokovne literature, ki sta Kwon-Zmud-Cooperjev 6-stopenjski model sprejetja inovacije in Rogersova teorija difuzije inovacije. Podatke za analizo smo zbrali s pomočjo vprašalnikov in delno strukturiranih intervjujev. S pomočjo teh modelov smo identificirali faktorje, ki imajo pri dolgoročnem sprejemanju metodologije Scrum največji vpliv. Nanje se je treba osredotočiti pri izvajanju projektov v prihodnje, če želimo obdržati razvoj po metodologiji Scrum.

S postmortem analizo smo članom skupine Scrum omogočili, da so se še enkrat ozrli nazaj in skupaj ugotovili, kaj je bilo dobro in kaj bi bilo smiselno spremeniti, da se izognemo podobnim napakam v prihodnje. Ugotovili smo, da se podjetje v prihodnje lahko izogne težavam, ki so bile zaznane na obravnavanem projektu, in sicer pred pričetkom novega projekta po metodologiji Scrum s skrbno izbrano skupino Scrum in načrtovanjem vključitve naročnika ob začetku projekta. Med samim projektom skupina Scrum ne sme pozabiti na spremljanje napredka razvoja in uporabi tehniko za ocenjevanje uporabniških zgodb, ki je dovolj preprosta za razumevanje in uporabo ter prinaša dobre rezultate.

V nadaljnjih korakih bi bilo zanimivo raziskati, kateri faktorji sprejemanja inovacije vplivajo na sprejem Scruma pri tistih članih skupine Scrum, ki se z njim srečujejo prvič in kako pri tistih, ki imajo z njim že pozitivne izkušnje. Zanimalo bi nas, ali na ti dve skupini ljudi pri sprejemljivosti Scruma vplivajo isti ali različni faktorji. Rezultati take študije bi nam pomagali pri nadaljnjih projektih, kjer bi se srečevali s podobno strukturo članov skupine Scrum glede predhodne izkušenosti oziroma kako oblikovati skupine Scrum, kjer je del članov že izkušenih, del pa ne, kar je vsekakor dejstvo, če želimo Scrum razširiti po celem podjetju.

Uporaba rezultatov tega dela je smiselna pri vsakem podjetju, ki uvaja Scrum in ga želi dolgoročno sprejeti. To delo nudi tudi ogrodje, po katerem lahko katerokoli podjetje, ki uporablja metodologijo Scrum, izvede študijo in preveri, kateri so faktorji, ki na njihovem primeru vplivajo na dolgoročno sprejemljivost Scruma. S pomočjo tega dela lahko preverijo sprejetost posameznih praks in izvedejo ustrezne ukrepe za izboljšanje tistih praks, ki so slabše sprejete, saj so za dobre učinke Scruma vse prakse pomembne.

Rezultati te študije so bili predstavljeni tretjemu skrbniku metodologije na obravnavanem projektu, kar je vsekakor dobrodošlo, saj je na ta način dobil povratno informacijo o svojem delu in spremembah, ki jih je uvedel. Bil je zelo pozitivno presenečen, kako zrelo o Scrumu razmišljajo razvijalci, ki so bili za potrebe te študije intervjuvani. Omenjali so prav te prednosti, ki jih Scrum omogoča, to so hitrejši razvoj, boljša koda, boljše vzdušje v skupini in večja pomoč med člani skupine. Poleg tega je dobil informacije, kje se še lahko izboljšamo in kje je treba vztrajati, da bomo tudi v prihodnje razvijali po metodologiji Scrum, kar si razvojna ekipa vsekakor želi. V prihodnje nameravamo rezultate te študije predstaviti tudi vodstvu podjetja, ki je izvajalo razvoj programske rešitve.

10 PRILOGE

10.1 DODATEK A

Prvi del vprašalnika je vseboval štiri splošna vprašanja anketiranca:

1. **V kakšni vlogi si nastopal na projektu prenove portala za državljane? V kolikor si nastopal v več vlogah, izberi tisto, ki ti je najbolj ustrezala.**

   a) Skrbnik metodologije (Scrum Master)
   b) Produktni vodja (Product Owner)
   c) Član razvojne skupine (Team Member)
   d) Drugo

2. **Koliko izkušenj že imaš z uporabo metodologije Scrum?**

   a) nimam še izkušenj z uporabo metodologije Scrum
   b) manj kot 1 leto
   c) 1 do 3 leta
   d) 3 do 5 let
   e) več kot 5 let

3. **Kdo so bili ScrumMasterji v času, ko si bil na projektu? (možnih je več odgovorov)**

   a) ScrumMaster 1[4]
   b) ScrumMaster 2
   c) ScrumMaster 3

4. **Kdo so bili Product Ownerji v času, ko si bil na projektu? (možnih je več odgovorov)**

   a) Product Owner 1[5]
   b) Product Owner 2

[4] V dejanskem vprašalniku so bila navedena dejanska imena skrbnikov metodologije.
[5] V dejanskem vprašalniku sta bili navedena dejanski imeni produktnih vodij.

10.2 DODATEK B

Drugi del vprašalnika je vseboval petnajst trditev o uporabi praks Scrum:

1. **Vzdrževanje seznama zahtev (Product Backlog).**
   Seznam zahtev (Product Backlog) je gonilo razvoja. Seznam zahtev ni nikoli dokončen in se ves čas projekta stalno dopolnjuje.

   a) Za **vzdrževanje seznama zahtev** slišim prvič.
   b) **Seznama zahtev** še nisem **vzdrževal.**
   c) Imam dovolj potrebnih znanj, da lahko **vzdržujem seznam zahtev**.
   d) Z **vzdrževanjem seznama zahtev** imam že pozitivne izkušnje.
   e) **Seznam zahtev** sem **vzdrževal** že tolikokrat, da je to zame postalo rutinsko opravilo.
   f) **Vzdrževanje seznama zahtev** izvajam rutinsko, učinkovito in celovito, kar olajša izvedbo ostalih ključnih aktivnosti Scruma in pohitri ter izboljša kakovost razvoja programske rešitve.

2. **Uporaba uporabniških zgodb (User Stories) za predstavitev zahtevane funkcionalnosti.**
   Namesto podrobne specifikacije zahtev, ki lahko »ubije« projekt, zadostuje kratek zapis zahtevane funkcionalnosti, ki služi kot opomnik, da vemo, o čem se je treba pogovarjati. Oblikovati je treba primerno velike uporabniške zgodbe, ki predstavljajo osnovo za ocenjevanje zahtevnosti in planiranje projekta.

   a) Za **uporabo uporabniških zgodb za predstavitev zahtevane funkcionalnosti** slišim prvič.
   b) **Uporabniških zgodb za predstavitev zahtevane funkcionalnosti** še nisem **uporabljal**.
   c) Imam dovolj potrebnih znanj, da lahko **uporabljam uporabniške zgodbe za predstavitev zahtevane funkcionalnosti**.
   d) Z **uporabo uporabniških zgodb za predstavitev zahtevane funkcionalnosti** imam že pozitivne izkušnje.
   e) **Uporabniške zgodbe za predstavitev zahtevane funkcionalnosti** sem **uporabljal** že tolikokrat, da je to zame postalo rutinsko opravilo.
   f) **Uporabniške zgodbe za predstavitev zahtevane funkcionalnosti uporabljam** rutinsko, učinkovito in celovito, kar olajša izvedbo ostalih ključnih aktivnosti Scruma in pohitri ter izboljša kakovost razvoja programske rešitve.

3. **Sodelovanje s produktnim vodjo (Product Owner) pri razčiščevanju podrobnosti uporabniških zgodb.**
   Podrobnosti, povezanih z realizacijo posameznih uporabniških zgodb, ne dokumentiramo, ampak jih razčiščujemo v pogovorih s Product Ownerjem.

   a) Za **sodelovanje s produktnim vodjo** slišim prvič.
   b) **S produktnim vodjo** še nisem **sodeloval.**
   c) Imam dovolj potrebnih znanj, da **sodelujem s produktnim vodjo.**
   d) S **sodelovanjem s produktnim vodjo** imam že pozitivne izkušnje.
   e) **S produktnim vodjo sem sodeloval** že tolikokrat, da je to zame postalo rutinsko opravilo.
   f) **S produktnim vodjo sodelujem** rutinsko, učinkovito in celovito, kar olajša izvedbo ostalih ključnih aktivnosti Scruma in pohitri ter izboljša kakovost razvoja programske rešitve.

4. **Skupinsko ocenjevanje uporabniških zgodb po metodi Planning Poker.**
   Metoda zagotavlja enakopravno sodelovanje vseh razvijalcev in onemogoča prevlado posameznikov.

   a) Za **ocenjevanje uporabniških zgodb po metodi Planning Poker** slišim prvič.
   b) **Uporabniških zgodb po metodi Planning Poker** še nisem **ocenjeval**.
   c) Imam dovolj potrebnih znanj, da lahko **ocenjujem uporabniške zgodbe po metodi Planning Poker**.
   d) Z **ocenjevanjem uporabniških zgodb po metodi Planning Poker** imam že pozitivne izkušnje.
   e) **Uporabniške zgodbe po metodi Planning Poker sem ocenjeval** že tolikokrat, da je to zame postalo rutinsko opravilo.
   f) **Uporabniške zgodbe po metodi Planning Poker ocenjujem** rutinsko, učinkovito in celovito, kar olajša izvedbo ostalih ključnih aktivnosti Scruma in pohitri ter izboljša kakovost razvoja programske rešitve.

5. **Skupinsko ocenjevanje uporabniških zgodb po metodi Team Estimation Game.**
   Metoda zagotavlja enakopravno sodelovanje vseh razvijalcev in onemogoča prevlado posameznikov.

   a) Za **ocenjevanje uporabniških zgodb po metodi Team Estimation Game** slišim prvič.
   b) **Uporabniških zgodb po metodi Team Estimation Game** še nisem **ocenjeval**.
   c) Imam dovolj potrebnih znanj, da lahko **ocenjujem uporabniške zgodbe po metodi Team Estimation Game**.

d) Z **ocenjevanjem uporabniških zgodb po metodi Team Estimation Game** imam že pozitivne izkušnje.
e) **Uporabniške zgodbe po metodi Team Estimation Game sem ocenjeval** že tolikokrat, da je to zame postalo rutinsko opravilo.
f) **Uporabniške zgodbe po metodi Team Estimation Game ocenjujem** rutinsko, učinkovito in celovito, kar olajša izvedbo ostalih ključnih aktivnosti Scruma in pohitri ter izboljša kakovost razvoja programske rešitve.

6. **Načrtovanje izdaje (Release Planning) na osnovi ocenjene hitrosti razvojne skupine.**
Uporabniške zgodbe razporedimo po sprintih glede na njihovo prioriteto. Seštevek točk vseh uporabniških zgodb v vsaki iteraciji mora biti enak ocenjeni hitrosti razvojne skupine. Pomembno je, da razvojna skupina čim bolj točno oceni svojo hitrost.

a) Za **načrtovanje izdaje** slišim prvič.
b) **Izdaje** še nisem **načrtoval**.
c) Imam dovolj potrebnih znanj, da lahko **načrtujem izdaje**.
d) Z **načrtovanjem izdaje** imam že pozitivne izkušnje.
e) **Izdajo sem načrtoval** že tolikokrat, da je to zame postalo rutinsko opravilo.
f) **Izdajo načrtujem** rutinsko, učinkovito in celovito, kar olajša izvedbo ostalih ključnih aktivnosti Scruma in pohitri ter izboljša kakovost razvoja programske rešitve.

7. **Načrtovanje vsebine naslednje iteracije (Sprint Planning).**
Načrtovanje naslednje iteracije (Sprint Planning) zahteva določitev ocene za hitrost (velocity), določitev uporabniških zgodb, ki jih bomo realizirali, razdelitev zgodb na naloge (tasks), določitev izvajalcev za posamezne naloge in ocenjevanje zahtevnosti nalog.

a) Za **načrtovanje vsebine naslednje iteracije** slišim prvič.
b) **Vsebino naslednje iteracije** še nisem **načrtoval**.
c) Imam dovolj potrebnih znanj, da lahko **načrtujem vsebine naslednje iteracije**.
d) Z **načrtovanjem vsebine naslednje iteracije** imam že pozitivne izkušnje.
e) **Vsebino naslednje iteracije sem načrtoval** že tolikokrat, da je to zame postalo rutinsko opravilo.
f) **Vsebino naslednje iteracije načrtujem** rutinsko, učinkovito in celovito, kar olajša izvedbo ostalih ključnih aktivnosti Scruma in pohitri ter izboljša kakovost razvoja programske rešitve.

8. **Vzdrževanje Sprint Backloga.**
   Med izvajanjem sprinta je potrebno dodajanje in (po potrebi) podrobnejše razčlenjevanje posameznih nalog ter beleženje količine vloženega in preostalega dela.

   a) Za **vzdrževanje Sprint Backloga** slišim prvič.
   b) **Sprint Backloga** še nisem **vzdrževal**.
   c) Imam dovolj potrebnih znanj, da lahko **vzdržujem Sprint Backlog**.
   d) Z **vzdrževanjem Sprint Backloga** imam že pozitivne izkušnje.
   e) **Sprint Backlog** sem **vzdrževal** že tolikokrat, da je to zame postalo rutinsko opravilo.
   f) **Sprint Backlog vzdržujem** rutinsko, učinkovito in celovito, kar olajša izvedbo ostalih ključnih aktivnosti Scruma in pohitri ter izboljša kakovost razvoja programske rešitve.

9. **Izvajanje Daily Scrum sestankov.**
   Ti sestanki omogočajo sproten vpogled v potek projekta in takojšnje ukrepanje v primeru težav.

   a) Za **izvajanje Daily Scrum sestankov** slišim prvič.
   b) **Daily Scrum sestankov** še nisem **izvajal**.
   c) Imam dovolj potrebnih znanj, da lahko **izvajam Daily Scrum sestanke**.
   d) Z **izvajanjem Daily Scrum sestankov** imam že pozitivne izkušnje.
   e) **Daily Scrum sestanke** sem **izvajal** že tolikokrat, da je to zame postalo rutinsko opravilo.
   f) **Daily Scrum sestanke izvajam** rutinsko, učinkovito in celovito, kar olajša izvedbo ostalih ključnih aktivnosti Scruma in pohitri ter izboljša kakovost razvoja programske rešitve.

10. **Uporaba diagrama Burndown za nadzor poteka dela.**
    Diagram prikazuje, kako se zmanjšuje količina neopravljenega dela. Uporablja se na ravni projekta oziroma izdaje kot celote (Release Burndown) in na ravni sprinta (Sprint Burndown). Razberemo lahko, kdaj bo projekt končan, oziroma ali bo delo opravljeno do konca sprinta.

    a) Za **uporabo diagrama Burndown** slišim prvič.
    b) **Diagrama Burndown** še nisem **uporabljal**.
    c) Imam dovolj potrebnih znanj, da lahko **uporabljam diagram Burndown**.
    d) Z **uporabo diagrama Burndown** imam že pozitivne izkušnje.
    e) **Diagram Burndown** sem **uporabljal** že tolikokrat, da je to zame postalo rutinsko opravilo.
    f) **Diagram Burndown uporabljam** rutinsko, učinkovito in celovito, kar olajša izvedbo ostalih ključnih aktivnosti Scruma in pohitri ter izboljša kakovost razvoja programske rešitve.

11. Dosledno upoštevanje koncepta »done«.

Vsaka uporabniška zgodba mora biti v celoti končana, da jo ob koncu sprinta lahko damo v uporabo. Rezultat vsakega sprinta mora biti *shippable fuctionality*.

a) Za **dosledno upoštevanje koncepta »done«** slišim prvič.
b) **Koncepta »done«** še nisem **dosledno upošteval**.
c) Imam dovolj potrebnih znanj, da lahko **dosledno upoštevam koncept »done«**.
d) Z **doslednim upoštevanjem koncepta »done«** imam že pozitivne izkušnje.
e) **Koncept »done«** sem **dosledno upošteval** že tolikokrat, da je to zame postalo rutinsko opravilo.
f) **Koncept »done« dosledno upoštevam** rutinsko, učinkovito in celovito, kar olajša izvedbo ostalih ključnih aktivnosti Scruma in pohitri ter izboljša kakovost razvoja programske rešitve.

12. Izvajanje Sprint Review sestankov.

Na Sprint Review sestankih skupina Scrum predstavi nove funkcionalnosti, ki so rezultat zadnjega sprinta, naročniku.

a) Za **izvajanje Sprint Review sestankov** slišim prvič.
b) **Sprint Review sestankov** še nisem **izvajal**.
c) Imam dovolj potrebnih znanj, da lahko **izvajam Sprint Review sestanke**.
d) Z **izvajanjem Sprint Review sestankov** imam že pozitivne izkušnje.
e) **Sprint Review sestanke** sem **izvajal** že tolikokrat, da je to zame postalo rutinsko opravilo.
f) **Sprint Review sestanke izvajam** rutinsko, učinkovito in celovito, kar olajša izvedbo ostalih ključnih aktivnosti Scruma in pohitri ter izboljša kakovost razvoja programske rešitve.

13. Izvajanje Sprint Retrospective sestankov.

Analiza dobrih in slabih strani dela v preteklem sprintu. Dogovor o izboljšavah v prihodnjih sprintih.

a) Za **izvajanje Sprint Retrospective sestankov** slišim prvič.
b) **Sprint Retrospective sestankov** še nisem **izvajal**.
c) Imam dovolj potrebnih znanj, da lahko **izvajam Sprint Retrospective sestanke**.
d) Z **izvajanjem Sprint Retrospective sestankov** imam že pozitivne izkušnje.
e) **Sprint Retrospective sestanke** sem **izvajal** že tolikokrat, da je to zame postalo rutinsko opravilo.

f) **Sprint Retrospective sestanke izvajam** rutinsko, učinkovito in celovito, kar olajša izvedbo ostalih ključnih aktivnosti Scruma in pohitri ter izboljša kakovost razvoja programske rešitve.

14. Poznavanje aktivnosti, ki jih izvajajo posamezne vloge Scruma (Product Owner, Scrum Master, Team).

Product Owner kot nosilec vizije, Scrum Master kot skrbnik metodologije in razvojna skupina, ki deluje po načelu samoorganizacije in kolektivno odgovarja za realizacijo.

a) Za **aktivnosti posameznih uporabniških vlog** slišim prvič.
b) Z **aktivnostmi posameznih uporabniških vlog se** še nisem soočil.
c) Imam dovolj potrebnih znanj, da **poznam aktivnosti posameznih uporabniških vlog**.
d) Z **aktivnostmi posameznih uporabniških vlog** imam že pozitivne izkušnje.
e) Z **aktivnostmi posameznih uporabniških vlog** sem **se soočil** že tolikokrat, da je to zame postalo rutinsko opravilo.
f) Z **aktivnostmi uporabniških vlog se soočam** rutinsko, učinkovito in celovito, kar olajša izvedbo ostalih ključnih aktivnosti Scruma in pohitri ter izboljša kakovost razvoja programske rešitve.

15. Uporaba mapiranja uporabniških zgodb (User Story Mapping) za gradnjo Product Backloga.

Mapiranje uporabniških zgodb zagotavlja strukturiran pristop pri planiranju izdaje (Release planning). Zagotavlja »big picture« sistema, kar omogoča učinkovito komunikacijo med člani skupine Scrum. Identificira pomanjkljivosti in odvečne funkcionalnosti v Product Backlogu.

a) Za **uporabo mapiranja uporabniških zgodb** slišim prvič.
b) **Mapiranja uporabniških zgodb** še nisem **uporabljal**.
c) Imam dovolj potrebnih znanj, da lahko **uporabljam mapiranje uporabniških zgodb**.
d) Z **uporabo mapiranja uporabniških zgodb** imam že pozitivne izkušnje.
e) **Mapiranje uporabniških zgodb** sem **uporabljal** že tolikokrat, da je to zame postalo rutinsko opravilo.
f) **Mapiranje uporabniških zgodb uporabljam** rutinsko, učinkovito in celovito, kar olajša izvedbo ostalih ključnih aktivnosti Scruma in pohitri ter izboljša kakovost razvoja programske rešitve.

10.3 DODATEK C

Tretji del vprašalnika se je nanašal na trditve o faktorjih Rogersove DOI teorije.

Tabela 14: Vprašalnik – trditve o skupini faktorjev inovacija

Faktor	Trditev
Relativna prednost	Metodologijo Scrum za razvoj programske opreme uporabljam, ker **je boljša** od metodologij razvoja programske opreme, ki sem jih uporabljal prej.
Preprostost uporabe	Metodologijo Scrum uporabljam, ker **je enostavnejša za uporabo** kot metodologije razvoja programske opreme, ki sem jih uporabljal prej.
Kompatibilnost	Metodologijo Scrum uporabljam, ker se veliko **bolj ujema z mojim želenim načinom dela, vrednotami in izkušnjami** kot metodologije, ki sem jih uporabljal prej.
Možnost opazovanja, kako inovacijo uporabljajo drugi	Metodologijo Scrum uporabljam, ker sem imel **možnost videti/opazovati**, kako druge skupine uporabljajo Scrum.
Možnost preizkušanja	Metodologijo Scrum uporabljam, ker jo je **moč zelo enostavno preizkusiti** v delovnem okolju.
Strošek	Metodologijo Scrum uporabljam, ker njena **uporaba ne povzroča dodatnih stroškov in napora**.
Razreševanje problema	Metodologijo Scrum uporabljam, ker **razrešuje veliko problemov** pri razvoju programske opreme, ki sem jih imel z uporabo prejšnjih načinov razvoja programske opreme.
Standard	Metodologijo Scrum uporabljam, ker postaja **standardna metodologija** v podjetjih za razvoj programske opreme.
Tehnološka prednost	Metodologijo Scrum uporabljam, ker je **veliko bolj napredna** od načinov razvoja programske opreme, ki sem jih uporabljal prej.

Tabela 15: Vprašalnik – trditve o skupini faktorjev naloga

Faktor	Trditev
Poslovna prednost	Metodologijo Scrum uporabljam, ker pomembno prispevka k **večjemu zadovoljstvu naročnika** s končno programsko rešitvijo in tako zagotavlja prednost na tržišču.
Prepoznavanje potrebe uporabnika	Metodologijo Scrum uporabljam, ker je **način dela**, ki ga predpisuje pri razvoju programskih rešitev, **skladen z mojimi potrebami**.
Odpor uporabnika	Metodologijo Scrum uporabljam, ker pri razvoju programskih rešitev **olajša izvedbo zahtevnejših nalog**.

Tabela 16: Vprašalnik – trditve o skupini faktorjev posameznik

Faktor	Trditev
Samostojno testiranje	Metodologijo Scrum uporabljam, ker lahko z njo **samostojno eksperimentiram**.
Mreža osebnih stikov	Metodologijo Scrum uporabljam, ker mi prijatelji in kolegi **večinoma zelo priporočajo** njeno uporabo.
Lastna pravila in nadzor nad lastnim delom	Metodologijo Scrum uporabljam, ker jo je mogoče **enostavno prilagoditi** mojemu načinu dela.
Učenje z delom	Metodologijo Scrum uporabljam, ker se je lahko **naučim in ovrednotim med praktičnim delom** na projektu razvoja programskih rešitev.

Tabela 17: Vprašalnik – trditve o skupini faktorjev okolje

Faktor	Trditev
Kulturne vrednote	Metodologija Scrum je **skladna z vrednotami okolja**, v katerem delam.
Tehnološka infrastruktura	Za uporabo metodologije Scrum **imamo na voljo vso potrebno tehnološko infrastrukturo**.
Pravila skupnosti	Metodologijo Scrum **uporabljamo točno tako, kot je predpisana v literaturi**.

Faktor	Trditev
Viri	Za uporabo metodologije Scrum **imamo na razpolago vse potrebne vire**, tj. literaturo, izobraževanje, čas za delo, dostop do Scrum coacha.

Tabela 18: Vprašalnik – trditve o skupini faktorjev organizacija

Faktor	Trditev
Medosebne mreže	Metodologijo Scrum uporabljam, ker **sodelavci**, ki jo že uporabljajo, večinoma **zelo priporočajo** njeno uporabo.
Vrstniške mreže	Metodologijo Scrum uporabljam, ker **programsko rešitev razvijam v skupini s sodelavci, s katerimi se tudi sicer veliko družim.**
Neformalna komunikacija	Metodologijo Scrum uporabljam, ker omogoča **veliko spontane in neformalne komunikacije** med člani razvojne skupine.
Tehnološke izkušnje	Metodologijo Scrum uporabljam, ker **imam z uporabo metodologij razvoja programske opreme veliko izkušenj.**
Delovne skupine	Metodologijo Scrum uporabljam, ker nam omogoča, da **sami načrtujemo delo in sami razrešimo večino problemov in težav**, do katerih pride med razvojem programske rešitve.
Mnenjski voditelji in zastopniki sprememb	Metodologijo Scrum uporabljam, ker **osebe, katerih mnenje spoštujem, večinoma zelo priporočajo uporabo** metodologije Scrum.
Soodvisnosti od drugih	Metodologijo Scrum uporabljam, ker **se z vsakim novim uporabnikom** te metodologije Scrum **znatno poveča njena koristnost za organizacijo**.
Tip posvojitelja	Metodologijo Scrum uporabljam, ker pogosto **začnem z uporabo novih tehnologij** prej kot moji prijatelji in kolegi.
Hierarhija upravljanja	**Zahteva podjetja,** da mora razvoj potekati po metodologiji Scrum, **je glavni razlog** da uporabljam to metodologijo.

10.4 DODATEK D

Četrti del vprašalnika se navezuje na hipoteze Overhage in Schlauderer.

Tabela 19: Vprašalnik – trditve o skupini faktorjev inovacija (Overhage- Schlauderer)

Faktor		Trditev
Relativna prednost	čas do trga	Metodologija Scrum omogoča **hitrejši razvoj programske opreme** od tradicionalnih načinov razvoja programske opreme.
	zahteve naročnika	Metodologija Scrum omogoča **razvoj programske opreme, ki bolj ustreza zahtevam naročnika,** od tradicionalnih načinov razvoja programske opreme.
	učinki učenja	Metodologija Scrum omogoča, da se pri **razvoju programske rešitve naučimo več** kot pri tradicionalnih načinih razvoja programske opreme.
	zadovoljstvo	Metodologija Scrum omogoča **večje zadovoljstvo razvijalcev z razvito programsko rešitvijo** od tradicionalnih načinov razvoja programske opreme.
Kompatibilnost	transparentnost	Metodologija Scrum omogoča **boljši pregled nad potekom razvoja programske rešitve** od tradicionalnih načinov razvoja programske opreme.
	sodelovanje	Metodologija Scrum omogoča **boljše sodelovanje razvijalcev pri razvoju programske rešitve** od tradicionalnih načinov razvoja programske opreme.
Kompleksnost	kompleksnost procesa	Metodologija Scrum **zmanjša kompleksnost procesa razvoja programske rešitve** v primerjavi s tradicionalnimi načini razvoja programske opreme.
	disciplina	Metodologija Scrum **zahteva več discipline pri razvoju programske rešitve** od tradicionalnih načinov razvoja programske opreme.

10.5 DODATEK E

Zadnji sklop vprašalnika vsebuje trditve v zvezi z delom na projektu.

Tabela 20: Vprašalnik – trditve o karakteristikah projekta

Značilnost	Trditev
Lociranost skupine	**Komunikacija in delo** skupine Scrum sta **veliko boljši**, če celotna skupina tekom razvoja programske opreme sedi **v isti sobi**.
Komunikacija	**Dobra komunikacija** med člani skupine Scrum **zelo vpliva na kakovost** programske rešitve.
Predstavitev razvijalcev	Kvaliteta kode je večja, če **razvijalci sami predstavljajo** implementirane **zgodbe** na Sprint Reviewju.
Hitrost razvijanja	**Pomanjkanje stalnega sodelovanja stranke** zelo **vpliva na hitrost** razvite rešitve.
Zadovoljstvo stranke	**Pomanjkanje stalnega sodelovanja stranke** zelo **vpliva na zadovoljstvo stranke** z razvito rešitvijo.
Usposobljenost ScrumMasterja	**Usposobljenost in osebna angažiranost ScrumMasterja sta ključni za uvajanje** metodologije Scrum.
Usposobljenost Product Ownerja	**Usposobljenost in osebna angažiranost Product Ownerja sta ključni za razvoj zgodb**, ki so prioritetni in zaključene celote, tj. shippable fuctionality.
KUDO box	Igra **KUDO box znatno prispeva k boljšemu skupinskemu delu** skupine Scrum.
Best Sprint Task Force	Igra **Best Sprint Task Force znatno prispeva k boljši samoorganiziranosti** skupine Scrum **in kvaliteti kode**.
Tabla Kanban	**Fizični Kanban board** je poleg elektronskega Kanban boarda, npr. Jira, koristen, saj prinaša **boljši pregled** nad trenutnim delom.
Ocenjevanje uporabniških zgodb	**Ocenjevanje zgodb v točkah**, tj. story points, **je** veliko **bolj objektivno** kot ocenjevanje v urah.
Menjava metodologije	V okolju, kjer je razvoj programskih rešitev izrazito tradicionalno usmerjen, tj. waterfall, je **preklop na metodologijo Scrum dolgotrajen**.
Sledenje metodologiji	V okolju, kjer je razvoj programskih rešitev izrazito tradicionalno usmerjen, tj. waterfall, **sledenje metodologiji Scrum terja veliko dodatnega napora**.
Osebna rast	Razvoj programske rešitve po metodologiji Scrum **je pozitivno vplival na mojo osebno rast in motivacijo** za delo.
Prihodnost	Želim si, da bi na naslednjem projektu razvoja programske rešitve, **spet razvijal po metodologiji Scrum**.

11 LITERATURA

[1] A. Birk, T. Dingsøyr, T. Stålhane, "Postmortem: Never Leavea Project without It", *IEEE Software*, št. 19, zv. 3, str. 43–54, 2002.

[2] M. Ceschi, A. Sillitti, G. Succi in S. De Panfilis, "Project management in plan-based and agile companies", *IEEE Software*, št. 22, zv. 3, str. 21–27, 2005.

[3] M. Cohn, *Succeeding with Agile: Software Development Using Scrum*, Michigan: Addison-Wesley, 2010.

[4] R. B. Cooper, R. W. Zmud, "Information technology implementation research: A technological diffusion approach", *Management Science*, št. 36, zv. 2, str. 123–139, 1990.

[5] N.K. Denzin, Y.S. Lincoln, *Handbook of Qualitative Research*, Thousand Oaks: Sage Publications, 2000.

[6] J. L. Eveleens, C. Verhoef, "The Rise and Fall of the Chaos Report Figures", *IEEE Software*, št. 27, zv. 1, str. 30–36, 2010.

[7] A. Fontana, J.H. Frey, *The Interview: From Structured Questions to Negotiated Text, Handbook of Qualitative Research*, Thousand Oaks: Sage Publications, 2000, str. 645–672.

[8] V. Kutnik, J. Dovžan, S. Vuksanović, "Manifest agilnega razvoja programske opreme", *InfoSRC*, št. 79, 2015.
Dostopno na: http://infosrc.editiondigital.com/infosrc-st-79#!manifest-agilnega-razvoja-programske-opreme (dostop 7.5.2016)

[9] T. H. Kwon, R. W. Zmud, Unifying the fragmented models of information systems implementation, v R. J. Boland, R. A. Hirschheim: *Critical Issues in Information Systems Research*, New York: Wiley & Sons, 1987, str. 227–251.

[10] V. Mahnič, "Introducing Scrum into the development of a news portal", *12th International Conference on Applied Informatics and Communications*, Istanbul, Turkey, avg. 2012, str. 109–114.

[11] V. Mahnič, T. Hovelja, "The Influence of Diffusion of Innovation Theory Factors on Undergraduate Students' Adoption of Scrum", *International Journal of Engineering Education,* 2016 (v tisku).

[12] G. Mangalaraj, R. Mahapatra and S. Nerur, "Acceptance of software process innovations – the case of extreme programming", *European Journal of Information Systems*, št. 18, zv. 4, str. 344–354, 2009.

[13] E. Mustonen-Ollila and K. Lyytinen, "Why organizations adopt information system process innovations: a longitudinal study using Diffusion of Innovation theory", *Information Systems Journal*, št. 13, zv. 3, str. 275–297, 2003.

[14] S. Overhage, S. Schlauderer, "Investigating the Long-Term Acceptance of Agile Methodologies: An Empirical Study of Developer Perceptions in Scrum Projects", v zborniku *45th Hawaii International Conference on System Sciences*, Maui, Hawaii, jan. 2012, str. 5452–5461.

[15] S. Overhage, S. Schlauderer, D. Birkmeier, J. Miller, "What Makes IT Personnel Adopt Scrum? A Framework of Drivers and Inhibitors to Developer Acceptance", v zborniku *44th Hawaii International Conference on System Sciences*, Kauai, Hawaii, jan. 2011, str. 1–10.

[16] J. Patton*, User Story Mapping Discover the Whole Story, Build the Right Product*, O'Reilly Media, 2014.

[17] L. Rising, N. S. Janoff, "The Scrum Software Development Process for Small Teams", *IEEE Software*, št. 17, zv. 4, str. 26–32, 2000.

[18] E. M. Rogers, *Diffusion of Innovations*, New York: Free Press, 2003.

[19] K. S. Rubin, *Essential Scrum: A Practical Guide to the Most Popular Agile Process*, Michigan: Addison-Wesley, 2012.

[20] M. Senapathi and A. Srinivasan, "Understanding post-adoptive agile usage: An exploratory cross-case analysis", *Journal of Systems and Software*, št. 85, zv. 6, str. 1255–1268, 2012.

[21] K. Schwaber, *Agile Project Management with Scrum*, Redmond: Microsoft Press, 2004.

[22] K. Schwaber, M. Beedle, *Agile Software Development with Scrum*, Upper Saddle River: Prentice Hall, 2002.

[23] K. Schwaber, J. Sutherland, *The Scrum Guide*, 2013.
Dostopno na: http://www.scrumguides.org/docs/scrumguide/v1/scrum-guide-us.pdf (dostop 28.4.2016)

[24] H. Takeuchi, I. Nonaka, "The new new product development game", *Harvard Business Review*, št. 64, zv. 1, str. 137–146, 1986.

[25] L. G. Tornatzky, K. J. Klein, "Innovation characteristics and innovation adoption-implementation: A meta-analysis of findings", *IEEE Transactions on Engineering Management*, št. 29, zv. 1, str. 28–45, 1982.

Ostali viri

[26] K. Beck in ostali, *Manifesto for Afile Software Development*, 2001.
Dostopno na: www.agilemanifesto.org (dostop 11.6.2016)

[27] S. W. Ambler, Has Agile Peaked?, *Dr. Dobb's Journal*, maj 2008.
Dostopno na: http://www.drdobbs.com/architecture-and-design/has-agile-peaked/207600615 (dostop 7.5.2016)

[28] K. Cedrone, *A Scrum Master's Guide to Effective Agile Retrospectives*, avg. 2015.
Dostopno na: https://www.velir.com/blog/2015/08/13/scrum-master-s-guide-effective-agile-retrospectives (dostop 1.5.2016)

[29] J. Grenning, *Planning poker or How to avoid analysis paralysis while release planning*, 2002.
Dostopno na: https://sewiki.iai.uni-bonn.de/_media/teaching/labs/xp/2005a/doc.planningpoker-v1.pdf (dostop 7.5.2016)

[30] Kudo Box & Kudo Cards: *How to nurture intrinsic motivation*,
Dostopno na: https://management30.com/product/workouts/intrinsic-motivation/ (dostop 30.4.2016)

[31] M. Lant, *Estimating Effort For Your Agile Stories*, 2010.
Dostopno na: https://michaellant.com/2010/07/05/estimating-effort-for-your-agile-stories-2/ (dostop 2.6.2016)

[32] VersionOne, *10th Annual State of Agile Report*, apr. 2016.
Dostopno na: https://versionone.com/pdf/VersionOne-10th-Annual-State-of-Agile-Report.pdf (dostop 9.4.2016)

[33] The Standish Group, *Chaos*.
Dostopno na: https://net.educause.edu/ir/library/pdf/ncp08083b.pdf
(dostop 22.5.2016)